UNIVERZITA PARDUBICE

DOPRAVNÍ FAKULTA JANA PERNERA

BAKALÁŘSKÁ PRÁCE

2020 František Šilar

Univerzita Pardubice
Dopravní fakulta Jana Pernera

Software pro zpracování výsledků z časomíry pro požární sport

Bakalářská práce

2020 František Šilar

Univerzita Pardubice
Dopravní fakulta Jana Pernera
Akademický rok: 2019/2020

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**František Šilar**
Osobní číslo:	**D16149**
Studijní program:	**B3709 Dopravní technologie a spoje**
Studijní obor:	**Aplikovaná informatika v dopravě**
Téma práce:	**Software pro zpracování výsledků z časomíry pro požární sport**
Zadávající katedra:	**Katedra informatiky v dopravě**

Zásady pro vypracování

Vytvořte program, který bude naměřené časy z časomíry odesílané po sériové lince ukládat do databáze výsledků v pořadí dle zadané startovní listiny. Program by neměl při měření vyžadovat žádnou, nebo jen minimální obsluhu časoměřiče.
Práce se bude zabývat automatickým zpracováním výsledků z časomíry pro požární sport. Bude řešit samotnou komunikaci mezi časomírou a počítačem, v další části se bude zabývat samotným vyhodnocením naměřených časů a umožní vytvořit výsledkové listiny pro soutěžící. Program umožní několik typů měření (jedna dráha, více drah) podle aktuálního nastavení samotné časomíry.
Program bude určen pro OS Windows a bude napsán v jazyce C#

Rozsah pracovní zprávy: **30 normostran**
Rozsah grafických prací:
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. DEITEL, Harvey M. C# for experienced programmers. Upper Saddle River, NJ: Prentice Hall, c2003. ISBN 0130461334.
2. JAVŮREK, Milan a Ivan TAUFER. Vyhodnocování experimentálních dat: výběr základních statistických metod. 2. doplněné a opravené vydání. Pardubice: vlastním nákladem, 2018. ISBN 978-80-270-3611-0.
3. PACÁKOVÁ, Viera. Štatistické metódy pre ekonómov. Bratislava: Iura Edition, 2009. Ekonómia. ISBN 978-80-8078-284-9.

Vedoucí bakalářské práce: **Ing. Zdeněk Drvota**
Katedra informatiky v dopravě

Datum zadání bakalářské práce: **28. listopadu 2019**
Termín odevzdání bakalářské práce: **31. ledna 2020**

L.S.

doc. Ing. Libor Švadlenka, Ph.D.
děkan

doc. Ing. Karel Greiner, Ph.D.
vedoucí katedry

V Pardubicích dne 28. listopadu 2019

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 7/2019 Pravidla pro odevzdávání, zveřejňování a formální úpravu závěrečných prací, ve znění pozdějších dodatků, bude práce zveřejněna prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 29. 05. 2020

František Šilar

PODĚKOVÁNÍ

Rád bych těmito řádky poděkoval všem, kteří mi při tvorbě bakalářské práce, byť jen malým, ale přesto pro mě cenným dílkem, pomohli. Především bych pak chtěl poděkovat Ing. Liboru Valeši za poskytnutí popisu způsobu komunikace časomíry přes sériovou linku a Ing. Pavlu Suldovskému za cenné rady při návrhu některých algoritmů softwaru. V neposlední řadě také děkuji svojí rodině za podporu.

ANOTACE

Bakalářská práce se zabývá procesem automatického zpracování sportovních výsledků z hasičské časomíry při měření disciplín požárního sportu. V práci jsou popsány jednotlivé disciplíny požárního sportu a způsoby měření těchto disciplín. Další část se zaměřuje na způsoby komunikace osobního počítače s jinými zařízeními. Následující kapitola se věnuje technologiím použitým při implementaci softwaru, jenž je zároveň praktickou částí bakalářské práce. Tato aplikace bude určena ke zpracování naměřených výsledků z časomíry LV 4,66 určené pro požární sport, jejíž výrobcem je Ing. Libor Valeš. Samotným vývojem a obsluhou tohoto programu se zabývá poslední část práce.

KLÍČOVÁ SLOVA

požární sport, vývoj aplikací, relační databáze, sériová komunikace, RS-232, Windows Forms, .NET, SQL, C#

TITLE

Software designed for processing the results from the chronometer for the fire fighting sports timing

ANNOTATION

The bachelor thesis deals with automatic processing of sports results gained from the fire brigade chronometric instruments during the timing of the disciplines of fire-fighting sports. The thesis describes the individual fire-fighting sport disciplines as well as the means of their timing. Next part of the thesis focuses on ways of communication of personal computer with other devices. The following chapter deals with the technologies used for the implementation of the software that forms the practical part of the bachelor thesis. This application will be designed to process the results measured with the LV 4,66 chronometer designed for fire-fighting sports, the manufacturer of which is Ing. Libor Valeš. The final part of the thesis deals with the process of the development and maintenance of this program.

KEYWORDS

fire fighting sports, application development, relational database, serial communication, RS-232, Windows Forms, .NET, SQL, C#

OBSAH

SEZNAM ILUSTRACÍ A TABULEK

SEZNAM ZKRATEK A ZNAČEK

DB	Relační databáze
GUI	Grafické uživatelské rozhraní
HZS	Hasičský záchranný sbor
MSSQL	Microsoft SQL Server
NP	Neplatný pokus
OS	Operační systém
PC	Osobní počítač
PHP	Přenosný hasící přístroj
PS	Požární sport
PÚ	Požární útok
SDH	Sbor dobrovolných hasičů
SSMS	Microsoft SQL Server Management Studio
SW	Software, programové vybavení počítače
VS	Microsoft Visual Studio

ÚVOD

Sport byl od svých prvopočátků vnímán především jako souboj. Za dob antického Řecka bylo v klání důležité především to, kdo koho porazí a jakým způsobem zvítězí. Až s příchodem moderní éry se objevila myšlenka bojovat s časem. Ale jak tento parametr s co největší přesností zaznamenávat? Na počátku byla všechna důvěra svěřena lidskému postřehu. Časy měřili rozhodčí se stopkami. Bývalo běžné, že pokus jednoho sportovce obvykle měřilo i několik rozhodčích, přičemž získané hodnoty se pak průměrovaly. Je tedy zřejmé, že ruční měření mělo zásadní chybu – postrádalo přesnost. Přestože lidské oko vidí velmi přesně, zásadním problémem je reakce člověka na zrakový vjem, jež bývá značně zpožděná. S rostoucími technologickými možnostmi lidstva a stále vyrovnanějšími výkony sportovců tak došlo i na automatizaci měření sportovních výsledků – byla vynalezena elektronická časomíra. V dnešní době tak není problém zaznamenávat výkony sportovců s přesností na tisícinu sekundy, přičemž tyto výsledky jsou online vyhodnocovány, či jinak porovnávány. Jen těžko si dnes dovedeme představit, že bychom sledovali v televizi závody, kde by nás příslušná grafika neinformovala o časech sportovců, a to nejen těch výsledných, ale i průběžných – např. za jednotlivá kola, jednotlivé úseky atp.

Již od dětství jsem členem SDH Horní Čermná. Za několik let jsem navštívil mnoho hasičských závodů, kde jsem se mohl setkat s oběma způsoby měření. Jako mladý hasič jsem zažil několik závodů, kde měření časů bylo prováděno ručně stopkami, zatímco v dnešní době je na každém větším závodě zaznamenávání sportovních výsledků elektronickou časomírou takřka samozřejmostí. I proto se před nedávným časem rozhodl SDH Horní Čermná elektronickou časomíru, především pro tréninkové účely, zakoupit. Avšak jak se později ukázalo, časoměřičů pro požární sport není v dnešní době dostatek. Důkazem nám bylo několik poptávek po měření některých klání. Sám jsem několik závodů, na které mi bylo umožněno nahlížet „z opačné strany," navštívil. A právě to mě přinutilo se na problematiku zaznamenávání a vyhodnocování sportovních výsledků zaměřit trochu více…

První část práce seznamuje čtenáře s hasičským sportem. Je zde vysvětlen samotný pojem „hasičský sport" a dále jeho dělení. V kapitole je podrobněji popsán především požární sport, což je odvětví hasičského sportu, na jehož disciplíny se bakalářská práce v dalších kapitolách odkazuje.

Druhá kapitola pojednává o měření disciplín požárního sportu. Je věnována především elektronickému zaznamenávání sportovních výsledků. Seznamuje čtenáře s disciplínami PS

z pohledu časoměřiče, popisuje způsoby vyhodnocení jednotlivých disciplín a představuje samotnou elektronickou časomíru a některé její prvky, jež se při měření používají.

Ve třetí části je bakalářská práce zaměřena na komunikaci mezi PC a jiným zařízením. Stručně shrnuje technologie přenosu informací u běžně používaných komunikačních rozhraní a také síťovou komunikaci prostřednictvím technologie Ethernet.

Čtvrtá kapitola shrnuje technologie použité k vývoji SW pro zpracování výsledků z elektronické časomíry.

Poslední kapitola se pak věnuje návrhu a samotné implementaci aplikace pro zaznamenávání a vyhodnocování výsledků z časomíry pro hasiče. Software se skládá ze tří modulů. První modul zajišťuje přenos a zpracování dat z elektronické časomíry do PC, druhý modul je propojen s prvním a zajišťuje ukládání naměřených časů do databáze včetně jejich přiřazení k daným soutěžícím. Dále umožnuje vytvářet startovní a výsledkové listiny jednotlivých závodů. Třetí modul je pak určen ke správě ostatních údajů uložených v tabulkách databáze. Zmíněný program je zároveň praktickou částí bakalářské práce.

1 HASIČSKÝ SPORT A JEHO DISCIPLÍNY

Hasičský sport je souhrnné označení pro několik disciplín souvisejících s prvky z práce profesionálního hasiče, které se dále dělí na další poddisciplíny. Disciplíny hasičského sportu buď věrně napodobují situace, kterým musí profesionální hasič čelit při opravdovém zásahu, nebo jen spojují některé pracovní úkony hasičského povolání s jinými sportovními odvětvími, zejména pak s atletikou. Hasičský sport dělíme na tyto disciplíny:

- Požární sport
- Klasické disciplíny CTIF
- Disciplíny TFA
- Hra Plamen – soutěž mladých hasičů
- Celoroční soutěž dorostu

Ač časomírou, jež je určena pro požární sport, lze měřit i některé z poddisciplín ostatních soutěžních oborů hasičského sportu, ať už proto, že si jsou tyto disciplíny velmi podobné, nebo je to pouze technicky možné, kapitola se jimi nezabývá. Požární sport si však nyní rozebereme podrobněji.

1.1 Požární sport

Požární sport spojuje některé atletické disciplíny s prací hasičů. V disciplínách PS je kladen důraz na obratnost, rychlost a nebojácnost soutěžícího. Dobrý závodník musí umět především překonat sám sebe a nesmí se bát ohně ani výšek. Požární sport se skládá ze čtyř disciplín:

- Běh na 100 m s překážkami
- Výstup do 4. podlaží cvičné věže
- Štafeta 4 x 100 m s překážkami
- Požární útok

Zatímco první dvě disciplíny jsou týmové, dvě zbývající jsou individuální. [1]

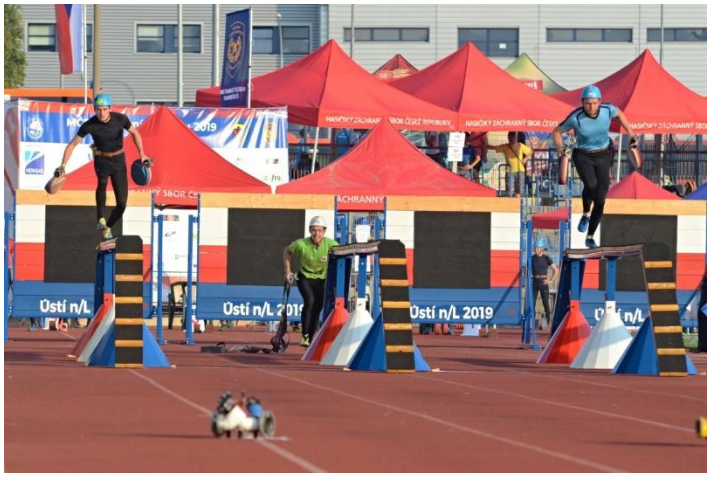
Požární sport se do České republiky rozšířil z bývalého Sovětského svazu, kde jej tamní hasiči provozovali již od roku 1937 hlavně pro zlepšení svých fyzických schopností a rychlosti při opravdovém zásahu. Poznatky o PS do našich končin přivezl odtamtud v roce 1967 Ing. Pavel Stoklásek a stal se tak jedním z největších propagátorů tohoto sportu u nás. V roce 1970 se požární sport stal součástí fyzické i odborné přípravy všech profesionálních hasičů tehdejší ČSSR, neboť byl zaveden do výkonu jejich služby. Od téhož roku jsou také každoročně pořádána mistrovství České republiky v PS. Dnes je požární sport pracovní náplní všech

profesionálních HZS podniků a HZS krajů. Od roku 1984 je mistrovství ČR společně pořádáno také pro dobrovolné hasiče. Ti však mají nepatrně odlišná pravidla. [1; 2]

1.1.1 Běh na 100 m s překážkami

Běh na 100 m s překážkami je individuální disciplína vycházející ze stejnojmenné lehkoatletické disciplíny. Závodník musí na trati co nejrychleji překonat 2 překážky a přitom spojit hadicové vedení, s jehož koncem dobíhá do cíle.

Soutěžící startuje s proudnicí, kterou může mít upevněnou za opaskem. Po odstartování překonává 2 m vysokou bariéru, první překážku. Ženy překonávají nízkou překážku (výška 80 cm). Následně závodník uchopí 2 hadice, s nimiž přebíhá přes kladinu. Pokud soutěžící z kladiny seskočí před čarou ohraničující vlastní dráhu, musí se vrátit na začátek a překážku překonat znovu. V opačném případě by nemohl být pokus započítáván jako platný. Výška vlastní kladiny je v mužské kategorii 120 cm, v kategorii „ženy“ 80 cm. Za druhou překážkou se nachází hadicový rozdělovač, na jehož spojku musí soutěžící připojit jeden konec zapojených dvou hadic, s nimiž tento běžec zdolal kladinu. Na druhý konec hadicového vedení závodník připojuje proudnici. Po tomto úkonu pak probíhá cílem. Pokud je některá ze spojek hadicového vedení špatně zapojena, případně drží-li soutěžící hadicové vedení tak, že rozhodčí nemůže správnost připojení před proběhnutím cílem posoudit, jedná se o pokus neplatný. [3; 4]



Obrázek 1 – Běh na 100 m s překážkami [30]

1.1.2 Výstup do 4. podlaží cvičné věže

Jedná se fyzicky nejnáročnější, avšak pro diváky zřejmě nejatraktivnější, individuální disciplínu PS. Právě kvůli své vysoké obtížnosti v ní mohou soutěžit pouze profesionální hasiči. Cvičná věž je tvořena kovovým profilem, k němuž jsou připevněna prkna, ve kterých jsou

umístěna tři okna – první ve druhém a následně ve třetím i čtvrtém podlaží. Úkolem soutěžícího je pomocí jednohákového žebříku vystoupat okny až do nejvyššího patra.

Soutěžící startuje s jednohákovým žebříkem ve vzdálenosti 32,25 m od cvičné věže. Po odstartování k ní přiběhne, žebřík zavěsí za parapet okna ve druhém podlaží a vyšplhá po něm. Poté usedne na parapet a převěsí žebřík o patro výše. Následně se úkony sportovce dvakrát opakují – šplh po žebříku do výše umístěného podlaží, usednutí na parapet, převěšení žebříku o úroveň výše (krom nejvyššího podlaží). Pokus končí při doteku podlahy čtvrtého patra cvičné věže, v téměř jedenáctimetrové výšce nad zemí, oběma nohama závodníka, případně sepnutím kontaktu cílového zařízení elektronické časomíry. Ženy mají disciplínu ulehčenu. Končí svůj pokus obdobně jako muži, avšak již v podlaží druhém. [4; 5]

Obrázek 2 – Výstup do 4. podlaží cvičné věže [31]

1.1.3 Štafeta 4 x 100 m s překážkami

Tato štafeta vychází z lehkoatletické disciplíny „běh na 4 x 100 metrů.“ Účastní se jí tedy 4 závodníci, přičemž každý na svém 100 m dlouhém úseku musí překonat 1 překážku. Štafetovým kolíkem je proudnice. Stejně jako v lehkoatletické disciplíně se jeho předávání musí uskutečnit ve 20 m dlouhém předávacím území dílčího úseku. V případě špatné předávky je družstvo diskvalifikováno.

První závodník vybíhá se štafetovým kolíkem a sklápěcím žebříkem. Pomocí něho musí soutěžící zdolat překážku zvanou „domeček“ Při překonávání se musí dotknout plošiny této překážky a seskočit za čáru vymezující vlastní dráhu. V opačném případě se závodník musí vrátit a zdolání opakovat, jinak bude celé družstvo ze štafety diskvalifikováno. Posledním úkolem prvního soutěžícího je předání proudnice v území k tomu určeném běžci na úseku č. 2. Ženy mají překonávání překážky opět zjednodušeno. Se sklápěcím žebříkem nestartují, ale mají ho již před startem opřený o zmíněnou překážku. Ostatní pravidla jsou shodná s mužskou kategorií. [4; 6]

Obrázek 3 – Domeček – Štafeta 4 x 100 metrů s překážkami [32]

Závodník na druhém úseku překonává 2 m vysokou bariéru, jež je stejná jako u disciplíny „běh na 100 metrů s překážkami“ Ženy zde taktéž překonávají pouze nízkou překážku (viz 1.1.1). Následuje předání štafetového kolíku třetímu soutěžícímu. [4; 6]

Ten má za úkol se dvěma hadicemi přeběhnout kladinu a sestavit hadicové vedení, stejně jako u běhu na 100 m s překážkami. Ženy mají kladinu opět nižší (viz 1.1.1). Provedení úkonu se liší na konci třetího úseku, kde musí mít soutěžící hadicové vedení viditelně spojené před vběhnutím do pásma pro odpojení. Zde závodník proudnici odpojí od hadicového vedení a s ní vbíhá do úseku předávacího, v němž ji předá poslednímu běžci. [4; 6]

Závodník na čtvrtém, posledním, úseku má za úkol pomocí přenosného hasícího přístroje uhasit zapálenou hořlavou směs v nádrži. Po předání proudnice od závodníka na úseku č. 3 přiběhne k hasícímu přístroji, zprovozní jej a utíká s ním k nádrži, kde s jeho pomocí uhasí zapálenou kapalinu. V případě potřeby může soutěžící užít také rezervní PHP, jenž je umístěn vedle nádrže s hořlavinou. Po uhašení nesmí hasící přístroj zůstat v nádobě, neboť by to při nedodržení zmíněného pravidla vedlo k diskvalifikaci celého družstva. V kategorii žen a u družstev dobrovolných hasičů může být (a ve většině závodů bývá) hašení nahrazeno pouze přenesením PHP na označené místo či podložku. Zde musí hasící přístroj stát až do proběhnutí

soutěžícího cílem. Pokud spadne a běžec se nevrátí zpět PHP postavit, je družstvo ze štafety vyloučeno. [4; 6]

Obrázek 4 – Hašení zapálené hořlavé směsi – Štafeta 4 x 100 metrů s překážkami [33]

1.1.4 Požární útok

Požární útok se svojí charakteristikou nejvíce podobá činnosti výjezdových jednotek hasičských sborů při požáru. Bývá také často označován za „královskou disciplínu“ PS. Jedná se o soutěž týmovou, přičemž toto družstvo je složeno ze sedmi závodníků. Jejich úkolem je dopravit vodu z vodního zdroje pomocí sacího vedení, výtlačného hadicového vedení se dvěma útočnými proudy a motorové požární stříkačky do vzdálenosti 95 m od osy základny, na kterou si běžci před startem připraví potřebné nářadí. Ve zmíněné vzdálenosti jsou umístěny 2 terče s nádržemi na vodu. Úkolem dvou členů družstva (dále jen proudaři) je pomocí 2 proudnic upevněných na každém z konců hadicového vedení daného útočného proudu co nejrychleji naplnit otvorem v terči 10 l nádobu. Pokud proudař úkol úspěšně splní, signalizace zmíněného cílového prvku o nastalé skutečnosti obvykle nějakým způsobem informuje a u elektronického měření především zastaví čas časomíry tohoto terče. Pokus je považován za úspěšně ukončený po naplnění nádob deseti litry vody u obou z terčů. Vše však musí být provedeno nejpozději do 2 minut od startu.

Na přípravu potřebného nářadí, jež se skládá z požární stříkačky, sacích požárních hadic (savic), „hadic B,“ „hadic C,“ sacího koše a hadicového rozdělovače, má družstvo 5 minut. Potřebné náčiní si musí na základnu (obvykle dřevěnou desku) připravit tak, aby se žádná z věcí nedotýkala země a s výjimkou savic také nepřesahovala obrys základny. Motor požární stříkačky nesmí být před startem v chodu. Žádná z částí dopravního vedení se nesmí

spojkami vzájemně dotýkat a musí být zřetelné, že jsou všechny prvky tohoto vedení nesešroubovány. Poté se závodníci přesunou ke startovní čáře vzdálené 9 m od hrany základny. Po startu k ní přibíhají zpět, kde část týmu sestaví sací vedení od vodního zdroje k požární stříkačce, nastartuje ji a s její pomocí zavodňuje výtlačné dopravní vedení, které mezitím uspořádali zbylí členové družstva. Při sestavování sacího vedení se nesmí hrdlo savice dotknout hladiny vody, aniž by na něm nebyl našroubován sací koš. V opačném případě se jedná o neplatný pokus. Také při překročení hraniční čáry stříkání umístěné před terči některým z členů týmu, záměně útočných proudů a terčů (levý útočný proud „stříkal“ na pravý terč či naopak), nebo překročení výše zmíněného časového limitu na dokončení je NP udělen. [4; 7]

Obrázek 5 – Sestavování dopravního vedení – Požární útok [34]

Obrázek 6 – Nástřik terčů – Požární útok [35]

2 MĚŘENÍ DISCIPLÍN POŽÁRNÍHO SPORTU

Konečná pravidla pro danou soutěž upravují až tzv. „propozice," což je dokument, který je vydán pořadatelem závodu před jeho zahájením v souladu s pravidly PS, jejichž poslední revize je z roku 2018 (k r. 2020). Tato listina např. jasně definuje, pro které soutěžní kategorie je závod určen, v jakých disciplínách se bude závodit a také jak budou soutěžícím měřeny výsledné časy. Pravidla požárního sportu připouští dvě možnosti zaznamenávání časů závodníků, a to buď ručně (nejčastěji digitálními stopkami), anebo pomocí elektronické časomíry. Avšak u soutěží krajských kol, při mistrovstvích a mezinárodních závodech v PS předpisy použití elektronické časomíry nařizují. Tato pak musí měřit čas s přesností min. na dvě desetinná místa, nebo více. Rovněž národní rekordy v požárním sportu mohou být uznány pouze za předpokladu zaznamenání nejlepšího výkonu pomocí elektronické časomíry. Uvedená pravidla zároveň se stále vyrovnanějšími výkony sportovců ruční měření pomalu, ale jistě vytlačují. Není tak dnes výjimkou setkat se s časomírou i na regionálních soutěžích. Pojďme si nyní elektronické měření přiblížit podrobněji.

Elektronická časomíra se obvykle skládá z hlavní jednotky, jež zajišťuje samotné měření času, propojovací kabeláže a dále ze startovacích a cílových zařízení. Jak již názvy napovídají, startovací zařízení slouží ke spuštění měření času a cílová k jejímu zastavení. Dnešní časomíry jsou obvykle ještě vybaveny výstupy pro externí informační displej sloužící k zobrazování výsledných časů např. na soutěžích a také výstupem pro PC za účelem zpracování naměřených hodnot. Zařízení určená přímo pro požární sport pak mívají ještě vstupy pro startovací zařízení určené pro spuštění odpočtu přípravy k provedení pokusu, jež je pravidly PS, případně propozicemi daného závodu, časově omezena. [4]

Časomíry pro PS jsou nejčastěji vybaveny jedním vstupem pro startovací zařízení ke spuštění měření a min. dvěma vstupy pro cílová zařízení. Požární časomíry mají obvykle dva režimy provozu – režim měření požárního útoku a režim měření štafety. V následujících dvou podkapitolách si ukážeme, jak zaznamenávání časů v těchto módech probíhá.

2.1 Měření požárního útoku

Vše začíná přípravou nářadí soutěžního družstva na základně. Na povel rozhodčího si dané družstvo začíná chystat součásti pro sestavení na základnu. V tuto chvíli také rozhodčí spouští odpočet přípravy. Na mnoha závodech je tento čas stále měřen stopkami. Existují ale i časomíry, jež dokáží přípravu taktéž měřit. V tomto případě je pak odpočet spuštěn stisknutím

startovacího zařízení pro přípravu. Dojde-li k uplynutí stanoveného intervalu, závodníci jsou o této skutečnosti informováni. Poté již na základně nesmí cokoli upravovat či měnit a musí se neprodleně dostavit na startovní čáru.

2.1.1 Start pokusu

Start probíhá nejčastěji pomocí startovací pistole, jež má v sobě zabudovaný spínač. Ten, v závislosti na jeho druhu, reaguje na stisknutí spouště či tlakovou vlnu po výstřelu náboje, čímž dojde k sepnutí spínače a k odeslání impulsu do hlavní jednotky časomíry, jež odstartuje čas. Existují také snímače zvukové, které nebývají osazeny přímo v pistoli, ale startér ji k němu před výstřelem přiloží a čas časomíry je spuštěn po zaznění výstřelu. Na některých soutěžích bývá zahájení měření prováděno pomocí tzv. „optických závor“. To je zařízení skládající se ze dvou komponent, vysílače a přijímače. Obě jednotky jsou nasměrovány proti sobě a umístěny na koncích startovní čáry, každý prvek na jednom. Vysílač pak vysílá proti přijímači světelný paprsek. Při přerušení tohoto paprsku dojde k vyslání impulsu do časomíry a spuštění času. Závodníci pak reagují na předem smluvený signál ke startu a časomíru „spouští“ sám soutěžící, případně běžec, který nejrychleji odstartuje (u hromadného startu v týmových soutěžích).

Obrázek 7 – Startovací zařízení – Optická závora [36]

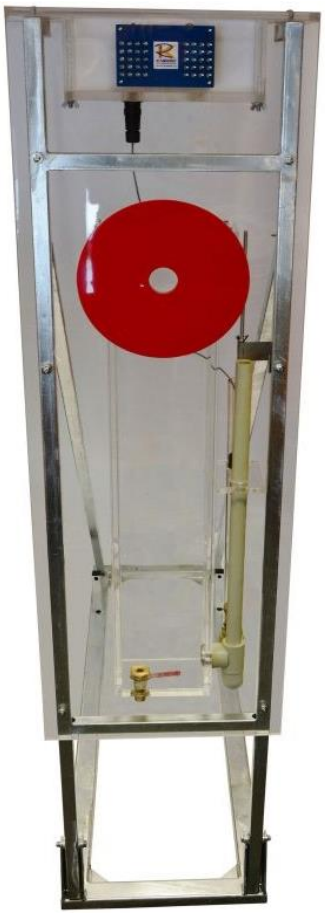
Obrázek 8 – Startovací zařízení – Startovací pistole [37]

2.1.2 Ukončení pokusu

Po odstartování pokusu hlavní jednotka časomíry očekává sepnutí koncových zařízení obou dvou terčů. Poté je zastaveno také měření času. K ukončení však může, ale nemusí dojít. Nebudeme-li brát v úvahu technickou závadu na časomíře a jejích prvcích, koncová zařízení nejsou sepnuta chybou soutěžního družstva. V takovém případě obsluha časomíry ukončí dané měření, čímž dojde k vynulování časomíry. Cílové prvky na terčích se mohou lišit. Pravidla PS

připouští pouze terče nástřikové, ve kterých se nachází hladinový spínač. Ten po naplnění terče deseti litry vody, kdy hladina v měrné nádobě přesáhne určitou mez, vyšle impuls do hlavní jednotky, čímž je čas terče zastaven. Existují však také jiné druhy terčů, jež se používají zejména na závodech soutěžních lig, které mají svá vlastní pravidla opírající se o předpisy požárního sportu. Jedná se zejména o terče pro plechovky, nebo především o terče „sklopné." Koncové zařízení v terčích pro plechovky pracuje na principu obyčejného vypínače se zapínacím kontaktem, na jehož ploše je nádoba umístěna a po jejím sražení dojde ke změně polohy tohoto spínače, což má za následek vyslání impulsu pro zastavení času.

Terč sklopný může být na první pohled nerozeznatelný od terče nástřikového. Liší se pouze tím, že místo jímací nádoby na vodu je za jeho otvorem umístěn tzv. válečkový mechanismus. Ten je při úspěšném pokusu proudem vody „sklopen", čímž dochází k zastavení času terče. Jako koncové zařízení se zde používá buď obyčejný přepínač, do jehož plochy po překlopení část mechanizmu „narazí" a dochází ke změně polohy vypínače, nebo modernější magnetický snímač. Ten po překlopení kovového válečku, kdy dochází k jeho dostatečnému přiblížení k senzoru, spíná.



Obrázek 9 – Nástřikový terč [38]

Obrázek 10 – Detail mechanismu sklopného terče [39]

2.2 Měření ostatních disciplín požárního sportu

Ostatní disciplíny požárního sportu probíhají v režimu provozu časomíry „štafeta“. Ta se od módu „požární útok“ liší počtem koncových zařízení potřebných k měření jednoho pokusu. Zde je pouze jedno toto zařízení, tedy po jeho sepnutí je čas časomíry ihned zastaven. Také příprava k provedení pokusu je měřena obdobně, buď stopkami, nebo pomocí časomíry (viz 2.1). Závodníci se však nepřipravují na základně, ale chystají si potřebné nářadí ve svém úseku na dané soutěžní dráze. Doba přípravy se liší v závislosti na konkrétních disciplínách a propozicích závodu. U týmové štafety 4 x 100 m s překážkami je to do 5 minut, u individuálních disciplín je to pak maximálně do 2 minut. Po této době musí být již soutěžící připraven na startu či v příslušném úseku na dráze.

2.2.1 Start pokusu

Start pokusu ostatních disciplín PS je obdobný jako v kapitole 2.1.1

2.2.2 Ukončení pokusu

Přestože je průběh ostatních disciplín PS z pohledu soutěžících narozdíl od požárního útoku zcela odlišný, z pohledu elektronického měření se ostatní disciplíny téměř neliší. Hlavní rozdíl, jak již bylo řečeno, je v počtu, ale také v typu samotných koncových zařízení. I zde však k jejich sepnutí vlivem chyby závodníka nemusí dojít.

U štafety 4 x 100 m s překážkami a u běhu na 100 m s překážkami se jako cílová zařízení nejčastěji používají optické závory. Toto zařízení je podrobněji popsáno v kapitole 2.1.1 Tam je však použito jako zařízení startovací sloužící ke spuštění měření času, zatímco zde je pochopitelně zapojeno jako koncové, tedy měření zastavuje.

U disciplíny Výstup do 4. podlaží cvičné věže se pak nejčastěji setkáme s následujícími dvěma typy koncových spínačů. Dle pravidel PS se používá tzv. nášlapná deska, ale na některých soutěžích se můžeme setkat i s obyčejným přepínačem, který soutěžící ve vrcholovém podlaží stiskne.

Nášlapná deska je pak koncové zařízení vybavené dvěma vypínači se zapínacími kontakty. Ty se nacházejí pod oběma nášlapnými plochami rozdělenými pevnou částí desky. Po dostatečném zatížení obou těchto ploch (soutěžící zatíží obě nášlapné plochy svojí vahou těla) dojde k sepnutí obou spínačů. V tento moment koncové zařízení vyšle signál k zastavení času do hlavní jednotky časomíry.

Obrázek 11 – Nášlapná deska pro výstup na věž [40]

2.3 Vyhodnocování disciplín požárního sportu

Máme-li časy soutěžících naměřeny, můžeme přistoupit k jejich vyhodnocení. Prvním a samozřejmě zásadním kritériem je pochopitelně zaznamenaný čas. V některých soutěžích požárního sportu má dle pravidel daný tým či závodník dva pokusy na konkrétní disciplínu. Na konci závodu tedy můžeme mít k jednomu soutěžícímu přiděleny dva výsledné časy. Avšak časové kritérium určující, jak budou tyto hodnoty vyhodnocovány, není ve všech disciplínách PS totožné. Další podkapitola obsahuje seznam ostatních hodnotících kritérií. Některé tyto rysy pravidla PS neuvádí, přesto se však s nimi na závodech můžeme setkat.

2.3.1 Časové kritérium

Požární útok

U požárního útoku je rozhodující čas pomalejšího terče. Ve výsledkových listinách bývá sice uveden i terč rychlejší, ale jeho hodnota nemá vliv na konečné pořadí. V případě více pokusů se jako rozhodující počítá čas pomalejšího terče v nejlepším z pokusů daného týmu. Pokud má více týmů shodný výsledný čas, budou všechna tato družstva taktéž shodně hodnocena. [4]

Běh na 100 m s překážkami, Výstup do 4. podlaží cvičné věže

U těchto disciplín v případě více pokusů o pořadí rozhoduje nejlepší čas ze všech pokusů závodníka. V případě týmové soutěže o pořadí rozhoduje součet šesti nejlépe umístěných nejrychlejších časů závodníků. [4]

Štafeta 4 x 100 m s překážkami

Zde o pořadí rozhoduje nejlepší pokus daného družstva s tím, že každý závodník může v této disciplíně soutěžit pouze jednou. [4]

2.3.2 Ostatní rozhodující kritéria

V následujících řádcích jsou uvedena další kritéria, podle kterých může být společně s kritériem časovým provedeno vyhodnocování. Uvedené možnosti mohou být také kombinovány.

Početní kategorie

Kritériem početní kategorie se rozumí, zda má být vyhodnocení časů provedeno pouze v rámci soutěžních týmů, nebo u každého závodníka individuálně.

Soutěžní kategorie

Závod může být vyhodnocen v rámci jedné soutěžní kategorie (např. muži), nebo mohou být soutěžní kategorie vyhodnocovány dohromady.

Soutěžní liga

Na některých závodech se můžeme setkat se situací, kdy je daná soutěž zároveň zařazena do jedné nebo více soutěžních lig. Avšak družstvo v některém z těchto soutěžních přeborů, nebo dokonce v žádném lize, nemusí soutěžit. Vyhodnocování je pak možné provádět buď v rámci jedné soutěžní ligy, nebo v rámci závodu, tedy se všemi soutěžícími účastnícími se tohoto klání.

Soutěžní kolo

Jedná-li se o soutěž, na jejíž disciplíny připadá více pokusů, mohou být výsledky, především pro zajímavost, vyhodnoceny za každé soutěžní kolo zvlášť. Jinak toto kritérium dle pravidel PS nemá vliv na výsledné pořadí.

2.4 Protesty a odvolání

Soutěžící má dle pravidel požárního sportu právo na podání protestů a odvolání se proti poškození nebo neregulérnímu vyhodnocení vlastního sportovního výkonu nebo výkonu jiného účastníka závodu. Protest se podává ústně nebo písemně prostřednictvím vzorového formuláře (příloha v pravidel PS) příslušnému rozhodčímu disciplíny nebo hlavnímu arbitrovi, který má povinnost se tímto nesouhlasem dále zabývat. Za složení protestu může být požadována kauce, jež je protestujícímu v případě uznání odvolání vrácena. Protesty se podávají v omezeném časovém limitu proti:

- **Provedení, průběhu, či hodnocení pokusu**, a to do 10 minut od ukončení sporného pokusu
- **Průběhu disciplíny**, a to do 10 minut od ukončení sporné disciplíny
- **Vyhlášení výsledků disciplíny**, a to do 10 minut od vyhlášení sporných výsledků této disciplíny

Výše uvedené protesty se podávají příslušnému rozhodčímu disciplíny. Dále lze podat odvolání vůči:

- **Povolení účasti soutěžícího v závodě**, a to před zahájením první disciplíny daného závodu
- **Vyhlášení výsledků soutěže**, a to do 10 minut od vyhlášení sporných výsledků této soutěže

Tyto protesty přijímá hlavní rozhodčí dané soutěže. [4]

3 KOMUNIKACE PC SE ZAŘÍZENÍMI

Komunikací se obecně rozumí výměna informací. S tímto pojmem se setkáváme denně, aniž si to možná uvědomujeme – Ať už jde o komunikaci lidskou, při které spolu hovoří osoby, nebo, poněkud modernější, technickou, kdy mezi sebou komunikují zařízení. Osobní počítače nejčastěji se zařízeními komunikují pomocí tří následujících rozhraní:

- Sériová linka (RS-232)
- USB
- Ethernet

Z dalších jsou to pak Wi-Fi a Bluetooth, technologie pro bezdrátovou komunikaci, nebo technologie FireWire, popsaná standardem IEEE 1394.

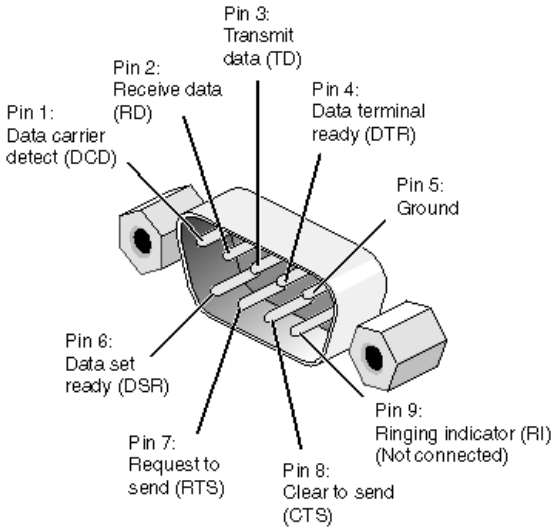
Časomíra, pro kterou byl software zpracovávající výsledky vyvíjen, je vybavena portem RS-232 sloužícím k odesílání naměřených časů do PC. Především z tohoto důvodu si sériovou komunikaci vysvětlíme podrobněji.

3.1 Komunikace prostřednictvím sériové linky (RS-232)

Sériová linka je nejstarším a také nejjednodušším způsobem přenosu informací mezi dvěma zařízeními. Standard pro sériovou komunikaci se nazývá RS-232C a byl popsán již v roce 1969. Informace se přenáší po bitech v sérii (postupně za sebou). Dříve byl sériový port hojně využíván. Dnes se již téměř nepoužívá a byl nahrazen výkonnějším USB. Přesto se s touto zastaralou technologií, zejména u kusové výroby elektrotechnických zařízení, jež umožňují komunikaci s PC, či u průmyslových zařízení, můžeme ještě setkat. Je to především pro již zmíněnou jednoduchost komunikace. A přestože většina dnešních počítačů a notebooků není sériovou linkou vybavena, mohou tato zařízení pomocí jejího standardu komunikovat. Existují totiž USB-RS232 převodníky. [8]

Pro obousměrný přenos dat stačí tři vodiče. Jeden pro vysílání (TD), druhý pro příjem (RD), a třetí vodič, společná zem, pro uzavření elektrického obvodu (Ground). Tento přenos je plně duplexní, 1 zařízení tedy může zároveň data přijímat i odesílat. Přenos obvykle probíhá jednou ze standardizovaných rychlostí – 4800 bitů za sekundu (b/s), 9600 b/s… Rozhraní

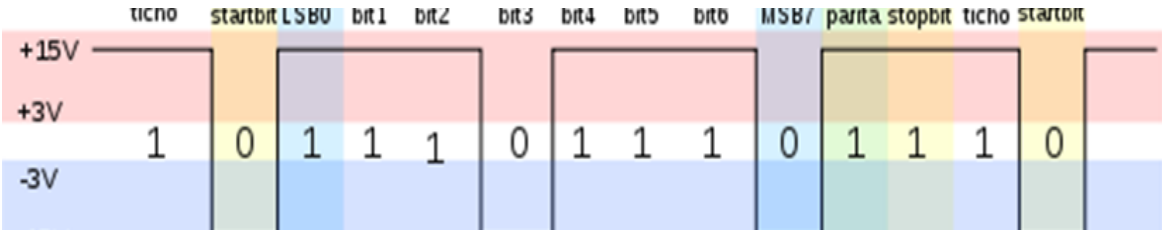
nadále ještě obsahuje další vodiče, jež slouží k řízení přenosu. Tyto piny můžou, ale nemusí být zapojeny. [8]



Obrázek 12 – Konektor sériového portu D-Sub DE-9 [41]

Přenos datových bitů po sériové lince probíhá od bitu nejméně významného (LSB) po bit nejvýznamnější (MSB). Použijeme-li asociaci se čtením čísla, bity se tedy posílají „zprava doleva." Komunikace probíhá asynchronně – přenos dat není řízen hodinovým signálem, který by určoval platnost vysílaných datových bitů. Komunikující zařízení proto musí předem znát a mít správně nastavenou rychlost přenosu. Přijímač pak musí data začít zpracovávat ve správný okamžik. Toho je docíleno pomocí tzv. „startbitů" a „stopbitů."

Neprobíhá-li žádná komunikace, je signál držen v klidové úrovni – v tomto případě „logická 1." Přenos dat je zahájen startbitem, jenž má opačnou úroveň oproti klidové – „log. 0." Pro přijímací zařízení tento bit značí „konec klidu" a to na něj reaguje spuštěním vlastního hodinového signálu. Jeho frekvence se odvíjí od nastavené přenosové rychlosti. Tento signál ovzorkuje přijímaná data, čímž dojde k určení platnosti hodnot datových bitů. Po odvysílání posledního (MSB) bitu následuje stopbit. Ten má logickou hodnotu shodnou s úrovní klidovou. Přijímač na něj reaguje přechodem do této úrovně a očekává příchod startbitu. Stopbit se tedy posílá nejen při ukončení komunikace, ale také po odvysílání nastaveného počtu datových bitů (obvykle 8). Je to především z důvodu synchronizace hodinového signálu přijímací stanice. Někdy je před stopbitem posílán také bit paritní, sloužící k detekci chyby ve slově. [8; 9]



Obrázek 13 – Sériová komunikace – Přenos 1 slova [42]

3.2 USB komunikace

USB je zkratka pro název Universal Serial Bus, neboli univerzální sériová sběrnice. Jedná se o moderní způsob připojení periferních zařízení k počítači. První verze USB komunikace, USB 1.0, byla poprvé standardizována v roce 1995. Od té doby byly vydány 4 revize (k r. 2020). Ta poslední, USB 3.1 Gen2, pochází z roku 2013 a disponuje teoretickou přenosovou rychlostí až 10 Gbit/s. To je téměř 834x vyšší rychlost, než kterou disponovalo USB 1.0. Přesto však zůstává zachována zpětná kompatibilita novějších verzí se staršími. Zařízení s různými revizemi USB rozhraní tak spolu dokáží komunikovat alespoň omezenou rychlostí přenosu. Další výhodou USB je podpora Plug & Play, tedy podpora připojení periferií za chodu bez nutnosti restartování počítače či instalování ovladačů. Zařízení je tedy po připojení prakticky ihned připraveno k použití. Není proto divu, že tato moderní sběrnice postupně nahradila všechna dříve používaná rozhraní pro připojování periferií k počítači. [10; 11]

Popis navázání USB komunikace je velmi složitý. Pro amatéra je nemožné vyrobit si vlastní zařízení, které by komunikovalo prostřednictvím USB. Každý výrobek používající tuto technologii navíc musí mít své ID a také ID výrobce. Pro přidělení těchto identifikujících hodnot je nutné podat žádost, které nemusí být vždy vyhověno. Jedinou rozumnou možností je tak koupě předpřipraveného hardwaru a softwarové knihovny. [10; 12]

3.3 Komunikace prostřednictvím technologie Ethernet

Ethernet je název pro jednu z technologií používaných v počítačových sítích. Jeho standard IEEE 802.3 byl poprvé popsán v roce 1980, přičemž v dalších letech byl ještě několikrát pozměněn a vylepšen. Ethernet je nejpoužívanější technologií v lokálních sítích (LAN), kde prostřednictvím společného komunikačního média propojuje jednotlivé síťové prvky. Těmi nejsou v dnešní době pouze počítače, ale i servery nebo chytré elektrické spotřebiče – televize, herní konzole atd. Jako přenosové médium se nejčastěji používá kroucená dvoulinka či optický kabel. V minulosti se pro propojování zařízení v síti používal také koaxiální kabel, avšak tato technologie je především pro nízkou přenosovou rychlost již zastaralá a překonaná. [13; 14]

S Ethernetem úzce souvisí také, v úvodu této kapitoly zmíněná, technologie Wi-Fi. Tu popisuje standard IEEE 802.11. Stejně jako Ethernet slouží v počítačových sítích k propojení zařízení v LAN. Komunikace je však bezdrátová. To je výhodou pro přenosná zařízení. Zároveň je ale nutné zmínit i zápory bezdrátové komunikace – jedná se především o větší náchylnost ke zneužití či menší odolnost vůči rušení oproti technologii „drátové.“ [14; 15]

4 TECHNOLOGIE POUŽITÉ K VÝVOJI APLIKACE

Seznamme se nyní s použitými technologiemi k vývoji softwaru pro zpracování výsledků z časomíry pro PS. K dodržení zásad pro vypracování uvedených v zadání bakalářské práce bylo nutné splnit tři podmínky: Vyhodnocování naměřených výsledků bude realizováno prostřednictvím databáze, software bude určen pro OS Windows a bude napsán v programovacím jazyce C#. Ostatní podmínky pro aplikaci výrobního postupu si tedy lze vybrat libovolně, neboť je zadavatel práce neuvádí. Tyto technologie byly vybrány především kvůli pozitivním zkušenostem z minulosti. Jde především o praxi nabytou při soukromých činnostech, anebo ze středoškolského či vysokoškolského studia.

4.1 .NET Framework

.NET Framework je jednou z implementací souboru softwarových technologií zvaných .NET. Tato programová kolekce je vyvíjena společností Microsoft a je určena k vývoji mnoha různých softwarových produktů. [16]

.NET Framework je základní komponentou zmíněného balíku, jež je optimalizována k vývoji desktopových aplikací pro OS Windows. Jedná se o nadstavbu tohoto operačního systému, která zvyšuje jeho bezpečnost a zajišťuje běh aplikací vyvinutých ve zmíněném frameworku. Jeho základem je Common Language Runtime (CLR). [17]

CLR obsahuje pravidla pro tvorbu překladačů jakéhokoli jazyka určeného pro prostředí .NET Framework, zásady pro definování a používání datových typů a také komponentu Garbage Collector, jež zajišťuje automatickou správu paměti dealokací objektů, na které již neexistuje žádný odkaz. To všechno mj. umožňuje běh a vzájemnou spolupráci aplikací napsaných v různých programovacích jazycích. [17]

Programy vyvíjené v .NET Frameworku se nepřekládají do strojového kódu, jak je tomu běžné například u jazyka C++, ale do mezijazyka Common Intermediate Language (CIL). K samotnému překladu instrukcí pro procesor dochází až teprve při spuštění programu, kdy se provede částečná kompilace aktuálně potřebných součástí aplikace. K překladům dalších částí kódu dochází až tehdy, je-li to nutné – tzv. Just-in-Time (JIT). [17]

Výhodou .NET frameworku je bezpečnost kódu, neboť výsledný strojový kód je optimalizován pro konkrétní počítač, na němž aplikace běží. Dalším obrovským kladem je množství knihoven, které framework obsahuje. Programátor tak nemusí znovu psát věci, jež se často používají, ale zároveň se v nich dá udělat mnoho chyb – např. řazení pole, implementace

některých datových struktur apod. Obrovským plusem je také multiplatformnost kódu – součásti sestavení můžeme využít nejen při vývoji aplikací pro Windows, ale také např. pro Linux, Mac OS či herní konzole XBox. Mezi nevýhody pak patří nutnost přítomnosti příslušné verze .NET Frameworku v zařízení, na němž má aplikace běžet, a také nepatrně větší paměťová náročnost a nižší výkon u některých druhů aplikací např. oproti stejné aplikaci psané C++ (avšak na úkor bezpečnosti). [17]

4.2 C#

C# je vysokoúrovňový objektově orientovaný programovací jazyk vytvořený společností Microsoft a přizpůsobený především pro technologie .NET, s nimiž je zároveň vyvíjen. Je založen na jazycích C++ a Java. C# je vhodný k tvorbě formulářových aplikací pro Windows, databázových programů, webových stránek či aplikací pro mobilní zařízení. [18; 19]

4.3 LINQ

LINQ je dotazovací jazyk integrovaný v .NET Frameworku, kde představuje jednotnou syntaxi pro přístup k datům, bez ohledu na jejich zdroj. Lze pomocí něj získávat zároveň data z databázového rozhraní, XML souboru nebo objektu v paměti. LINQ usnadňuje především třídění, filtrování a vyhledávání dat. Syntaxe tohoto dotazovacího jazyku se částečně podobá struktuře jazyka SQL. [20]

4.4 SQL

SQL je standardizovaný strukturovaný dotazovací jazyk, jenž se používá pro práci s daty v relačních databázích. Vývoj započal na začátku 70. let 20. století firmou IBM, která tehdy prováděla výzkum DB. Později k rozvoji jazyka přispěly i další firmy. O velký pokrok se postarala dnešní společnost Oracle Corporation, která na konci 70. let představila svoji platformu Oracle Database. Dnes standardy SQL implementuje téměř každá relační databáze, avšak obvykle nepodporuje všechny požadavky normy, nebo naopak obsahuje prvky, jež nejsou v normě obsaženy. Proto je přenositelnost SQL dotazů mezi jednotlivými DB omezená. [21]

Příkazy SQL můžeme rozdělit do 4 skupin. První soubor se používá k získání dat z DB. Druhá část slouží k vytváření struktur databáze. Do třetí skupiny patří příkazy pro nastavování přístupových práv a řízení transakcí. Všechny ostatní instrukce pak řadíme do 4. kategorie. [21]

4.5 Microsoft SQL Server Express 2019

Microsoft SQL Server je relační databázový a analytický systém. Je určen převážně pro e-shopy, byznys a řešení datových skladů. První verze byla vydána v roce 1988, tehdy ještě v režii firmy Sybase. Microsoft tento produkt v roce 1994 odkoupil a začal ho vyvíjet na vlastní náklady. V roce 2000 vyšla verze 8.0, jejíž jádro bylo kompletně přepsáno a značně vylepšeno. MSSQL vychází pravidelně vždy v několika edicích, z nichž je většina placená. [22]

Microsoft SQL Server Express 2019 je pak bezplatnou edicí určenou k vývoji a provozování desktopových, webových a malých serverových aplikací. Neplacená verze má však svá omezení. Její využití je redukováno pouze na 1 jádro procesoru a omezené množství operační paměti. Také velikost databáze je omezena, může mít maximálně 10 GB. Pro účely vývoje SW pro zpracování výsledků z časomíry je však verze Express dostačující. [23]

4.6 Microsoft SQL Server Management Studio 18

Microsoft SQL Server Management Studio je tzv. SQL Client software. Slouží nejen ke správě databází, ale i infrastruktury SQL. SSMS také obsahuje nástroje ke sledování instancí DB serveru a databází. Prostředí také umožňuje jednoduše vytvářet SQL dotazy a skripty. Aplikace je poskytována bezplatně a lze si ji stáhnout např. na webu Microsoft. [22; 24]

4.7 Microsoft Visual Studio Enterprise 2019

Microsoft Visual Studio je vývojové prostředí od společnosti Microsoft. Je primárně určené k vývoji konzolových i formulářových aplikací pro OS Windows, webových stránek a aplikací prostřednictvím platformy.NET. Avšak funkce Visual Studia lze rozšířit stažením rozšiřujících sad komponent. Pomocí VS tak můžeme například vyvíjet software určený pro mobilní aplikace, hry s využitím enginu Unity, programy pro jiné operační systémy či samotná rozšíření sady Visual Studio. Prostředí umožňuje kompilaci programů jak do strojového kódu, tak i do mezijazyka různých platforem – neboli tzv. řízeného kódu. Mezi základní součásti VS patří editor programového kódu, debugger sloužící k ladění kódu či designery pro tvorbu grafických aplikací, webu, nebo databázových schémat (Datasets). [25; 26]

Visual Studio je podobně jako bratrský Microsoft SQL Server vydáváno v několika edicích. První z nich, Community, je poskytována bezplatně, ale s omezenými funkcemi. Edice Professional a Enterprise jsou placené. Visual Studio Enterprise je plnohodnotnou a nejdražší sadou, Professional je pak jakousi „zlatou střední cestou“ mezi zbylými dvěma edicemi. [27]

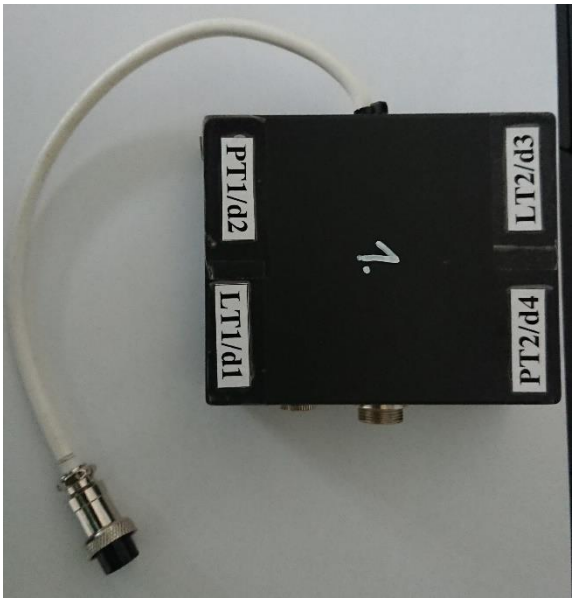
5 NÁVRH A IMPLEMENTACE ŘEŠENÍ

V této kapitole je shrnuta praktická část bakalářské práce. Na samém začátku je čtenář seznámen s elektronickou časomírou, pro niž byl software ke zpracování výsledků vyvíjen. Jsou zde charakterizovány základní součásti zařízení, popsány režimy měření a způsob komunikace po sériové lince mezi hlavní jednotkou a PC. Další část kapitoly je zasvěcena nově vyvíjené aplikaci. Seznámí nás se samotným návrhem součástí softwaru, a to nejen grafickým, ale i algoritmickým, který je nejdůležitějším prvkem celé aplikace. Přiblížíme si také databázový projekt, jež slouží nejen k ukládání naměřených sportovních výsledků, ale především k jejich vyhodnocení. Dále si řekneme pár věcí ohledně ovládání a nastavení aplikace. Závěr kapitoly je věnován instalaci a konfiguraci tohoto programu i dalších potřebných součástí.

5.1 Časomíra LV 4,66

Jedná se o časomíru vyráběnou Ing. Liborem Valešem, jež je primárně určena k měření disciplín požárního sportu. Je vybavena vstupem pro 1 startovací zařízení a umožňuje měřit zároveň až 2 dráhy požárního útoku, nebo až 4 dráhy ostatních disciplín PS. Napájení časomíry je stejnosměrné +12 V. Časomíru tedy lze napájet např. z autobaterie. Základní sestava se skládá z hlavní jednotky, rozbočovače pro koncová zařízení a propojovací kabeláže. Podle potřeby je možné u výrobce zakoupit také potřebná startovací i cílová zařízení (viz Měření disciplín požárního sportu). [28]

Obrázek 14 – LV 4,66 – Hlavní jednotka



Obrázek 15 – LV 4,66 – Rozbočovač pro koncová zařízení

5.1.1 Hlavní jednotka

Měření času zajišťuje reakcí na signály ze startovacích a cílových zařízení vnitřní elektronika hlavní jednotky. Toto zařízení obsahuje dvouřádkový LCD displej sloužící pro zobrazení naměřených výkonů a k nastavení časomíry. Samotná jednotka je relativně malých rozměrů, takže lze využít nejen na soutěžích, ale i pro tréninkové účely. Časomíra disponuje také možností odpočtu přípravy k provedení pokusu, avšak pouze pro 1 dráhu, či pro více drah současně, ale se stejným časem zahájení odpočítávání. Výsledné časy jsou měřeny s přesností na tři desetinná místa.

Ovládání

Časomíra se ovládá pomocí 1 tlačítka, které rozlišuje krátký a dlouhý stisk. Dlouhý stisk slouží především k vynulování naměřených časů, ale také k „vyvolání“ nastavení a „potvrzování možností“ v něm. Krátký stisk naopak použijeme pro spuštění odpočtu přípravy a v možnostech ke zobrazení další volby.

Konektory pro připojení součástí časomíry a jiných zařízení

Hlavní jednotka časomíry LV obsahuje v čelní části 3 vstupy pro konektory GX16. Vývody slouží k připojení startovacího zařízení, napájení a rozbočovače pro koncová zařízení. Po obou bocích jsou pak umístěny konektory D-Sub DE-9, na každém jeden. První z nich je určen k připojení informačního LED displeje, na kterém je zobrazován aktuální průběh měření. Druhým konektorem je vyvedeno sériové rozhraní RS-232 a slouží k přenosu dat z časomíry do PC.

Nastavení měření

Jak již bylo řečeno, časomíra LV umožňuje několik variant zaznamenávání časů. Disponuje dvěma základními režimy měření, u nichž lze dále nastavit počet drah, které mají být měřeny:

- **Požární útok** – Slouží k měření požárního útoku
  - 1 dráha
  - 2 dráhy
- **Štafeta** – Slouží k měření ostatních disciplín požárního sportu.
  - 1 dráha
  - 2 dráhy
  - 3 dráhy
  - 4 dráhy

Počet drah lze vybrat libovolně. Z důvodu omezeného vstupu pouze pro 1 startovací zařízení však musí mít v případě měření na více drahách tyto společný startovní signál. Nelze tak startovat každého závodníka individuálně např. pomocí optické závory, ale pouze všechny soutěžící jednotně, nejčastěji pomocí startovní pistole. U požárního útoku je ale tento způsob nevhodný. Startovní čáry jednotlivých drah jsou totiž od sebe poměrně vzdáleny. Startér tak musí stát mezi oběma drahami a pro některé týmy nemusí být v případě nepříznivých podmínek (vítr, náhlý hluk…) výstřel z pistole dobře slyšitelný. Další nevýhodou je také obtížnější kontrola předčasného startu členů obou týmů. Proto lze tento režim využít spíše při netradičních soutěžích – vyřazovacích (slangově „pavouk“) apod.

Nastavení doby přípravy

Dobu přípravy lze nastavit nezávisle na typu měření i počtu drah po 30 sekundách od 2 do 5 minut. Jak bylo zmíněno v úvodu této podkapitoly, tento časovač nelze nastavit pro každou dráhu individuálně, ale pouze jednotně a se stejným okamžikem zahájení odpočtu. To však na některých závodech nemusí být dostačující. Např. u požárního útoku je totiž běžné, že v případě závodu, kde soutěžení probíhá na více drahách, se družstvo na první dráze připravuje, zatímco na té druhé další tým přípravu ukončuje a chystá se k provedení výkonu či naopak. Možnost odpočtu přípravy tak lze u této časomíry využít pouze omezeně.

Ostatní nastavení

Hlavní jednotka dále umožňuje nastavit zobrazování běžícího času na LCD umístěných v prostoru cíle. Těmito displeji lze časomíru dovybavit. Dále je nutné pro správnou funkčnost v možnostech upřesnit typ cílových zařízení. Ta mohou být vybavena buď spínacími, nebo rozpínacími kontakty. Řídící jednotka tak reaguje buď na uzavření, nebo přerušení elektrického obvodu cílovým zařízením.

Koncová zařízení by měla být všechna stejného typu. V opačném případě může tato skutečnost způsobit nefunkčnost některých připojených druhů cílových jednotek, což se pochopitelně projeví neukončením měření na drahách s těmito odlišnými prvky.

5.1.2 Rozbočovač pro koncová zařízení

Rozbočovač se umísťuje v blízkosti cílových zařízení a slouží k jejich napojení na hlavní jednotku, se kterou je, nejčastěji prodlužovacím kabelem, propojen. Tento prvek časomíry LV 4,66 obsahuje 4 vstupy pro XLR konektory, jež slouží k připojení koncových prvků. Dále zde nalezneme 2 konektory GX16. Jeden je určen k propojení s řídící jednotkou a ten druhý

slouží jako výstup pro LCD displeje určené k zobrazení časů jednotlivých drah v prostoru cíle. V rozbočovači je také umístěn DC konektor pro připojení přídavného napájení LCD displejů a cílových součástí, je-li toto nutné např. v důsledku úbytku napětí na vedení (délka propojovací kabeláže je běžně i přes 100 m) mezi hlavní jednotkou a těmito zařízeními.

5.1.3 Sériová komunikace s hlavní jednotkou

Komunikace s PC je založena na odesílání příznakových bajtů po vzniku nějaké události vyvolané hlavní jednotkou. Spojení je tedy pouze jednosměrné (z řídící jednotky → PC). Nastavení sériového portu je standardní – Přenosová rychlost 9600 b/s, 8 datových bitů v 1 slově s jedním stopbitem na konci, bez parity. Nyní si přiblížíme výše zmíněné události a údaje, o kterých časomíra přes RS-232 informuje připojený počítač:

- Zahájení odpočtu přípravy
- Vynulování časomíry
- Zahájení měření časů
- Naměření výsledného času spolu jeho hodnotou a číslem dráhy
- Zobrazení posledních naměřených časů
- Změna nastavení časomíry spolu s novými hodnotami nastavení
- Změna připravenosti cílových zařízení k provedení dalšího pokusu

Informace o těchto událostech jsou definovány v jednoduchém komunikačním protokolu, jenž každé výše uvedené změně skutečnosti přiřazuje unikátní hexadecimální znak. Protokol dále obsahuje obdobná pravidla pro zasílání informací o nových hodnotách nastavení, o naměřených časech a také o sepnutí cílových zařízení dílčích drah. [29]

5.2 Představení aplikace Firesport Timekeeper

Program Firesport Timekeeper je realizován jako Multiple-Document Interface (MDI) aplikace. Základem je hlavní formulář, který slouží k ovládání podřízených formulářů. Výhodou MDI je možnost rychlého zobrazení více oken současně a také svižného přepínaní mezi nimi. K vytvoření podřízených formulářů bylo využito knihovny tříd Windows Forms, která je součástí .NET Frameworku a byla vyvinuta právě k tvorbě GUI pro desktopové počítače a Tablet PC.

5.2.1 Grafický návrh aplikace

Stěžejním bodem při implementaci řešení SW bylo navrhnout jeho grafické uživatelské rozhraní. To slouží především k interakci mezi uživatelem, ale i mezi jiným programem či zařízením, a danou aplikací.

Při vývoji grafického uživatelského rozhraní programu byl kladen důraz především na jednoduchost ovládání ukazatelem myši. Opomenuta však nebyla ani možnost rychlejší obsluhy pomocí klávesových zkratek. Samotná aplikace je pak rozdělena do 3 hlavních formulářů Windows Forms sloužících jako GUI tří „miniprogramů," které tvoří jednotlivé moduly výsledného softwaru. Každá ze zmíněných 3 součástí slouží k jinému účelu. Tyto záměry budou detailněji vysvětleny v jedné z dalších podkapitol. Nyní však pár slov ke grafickému návrhu dílčích formulářů.

Grafický návrh ovládacího formuláře

Tato část GUI zajišťuje zobrazování aktuálně potřebných grafických rozhraní jednotlivých modulů aplikace. Řídící formulář je složen z hlavního menu, které nabízí položky pro možnosti zobrazení podřízených formulářů, ale také hlavní nabídku aktuálně zobrazených modulů, jež se v případě jejich otevření s nabídkou řídícího formuláře sloučí. Prostor pod menu tohoto ovládacího formuláře slouží k zobrazení zmíněných modulů aplikace.

Grafický návrh modulu Časomíra

Tento modul zajišťuje především zpracování informací z časomíry a ukládání naměřených časů pro zálohu a další zpracování dat. Návrh Windows Forms formuláře byl proto zaměřený obzvláště na přehlednost zobrazovaných dat a také na dostatečnou informovanost uživatele o nastalých skutečnostech.

Ovládací prvky samotného formuláře můžeme rozdělit do 5 logických oddílů. V tom prvním se nachází hlavní nabídka, vytvořená pomocí ovládacího prvku *ToolStripMenuItem*, neboli roletkového menu s položkami. Mezi tyto údaje patří možnosti miniprogramu či nabídka připojení k sériovému portu. Dále to jsou volby pro import, export a editaci naměřených hodnot časů. Následující části jsou tvořeny ovládacími prvky *SplitContainer*. Jedná se o pohyblivý panel, jenž dělí zobrazovanou oblast kontejneru na dvě části. V těch najdeme další ovládací součásti, které již slouží k vizuálnímu předání informace obsluze programu.

V druhém oddílu nalezneme informace ohledně aktuálního nastavení časomíry, přípravy k provedení pokusu a také o sepnutí cílových zařízení po vynulování časomíry, pokud

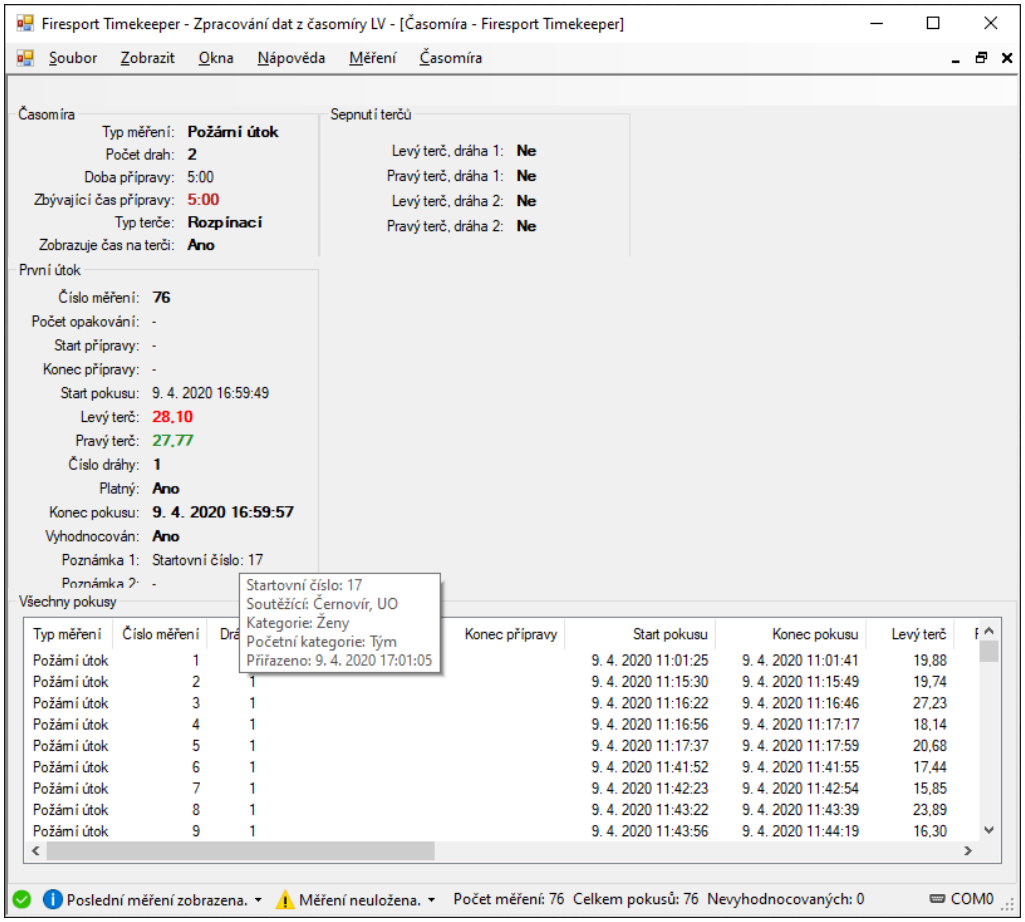
k takovému dojde. To je velmi vhodné, neboť hlavní jednotka časomíry v případě nepřipravenosti koncových zařízení nedovolí spustit další měření.

Třetí část zobrazuje aktuálně měřené časy nastavených drah. Najdeme zde přehled všech potřebných údajů o pokusech na daných tratích.

Ve čtvrtém oddíle nalezneme ovládací prvek *ListView*, ve kterém je zobrazena kolekce všech zaznamenaných časů. Tyto záznamy je možné, po vybrání právě jedné položky a zvolením příslušné možnosti v hlavním menu, editovat, případně k nim přidávat vlastní poznámky.

Posledním oddílem návrhu je stavový řádek, umístěný v dolní části formuláře. Na něm se zobrazují aktuální informace ohledně průběhu měření včetně statistik aktuální kolekce časů. Obsluze programu jsou zde zobrazena také případná varování, která mohou za běhu aplikace nastat. Nechybí ani informace o posledním uložení dat do souboru či o názvu sériového portu počítače, pomocí něhož program s časomírou komunikuje.

Některé ovládací součásti, které spolu logicky souvisí, jsou umístěny do ovládacího prvku *GroupBox*, který tyto prvky přehledně sdružuje, popisuje a ohraničuje.



Obrázek 16 – GUI modulu Časomíra

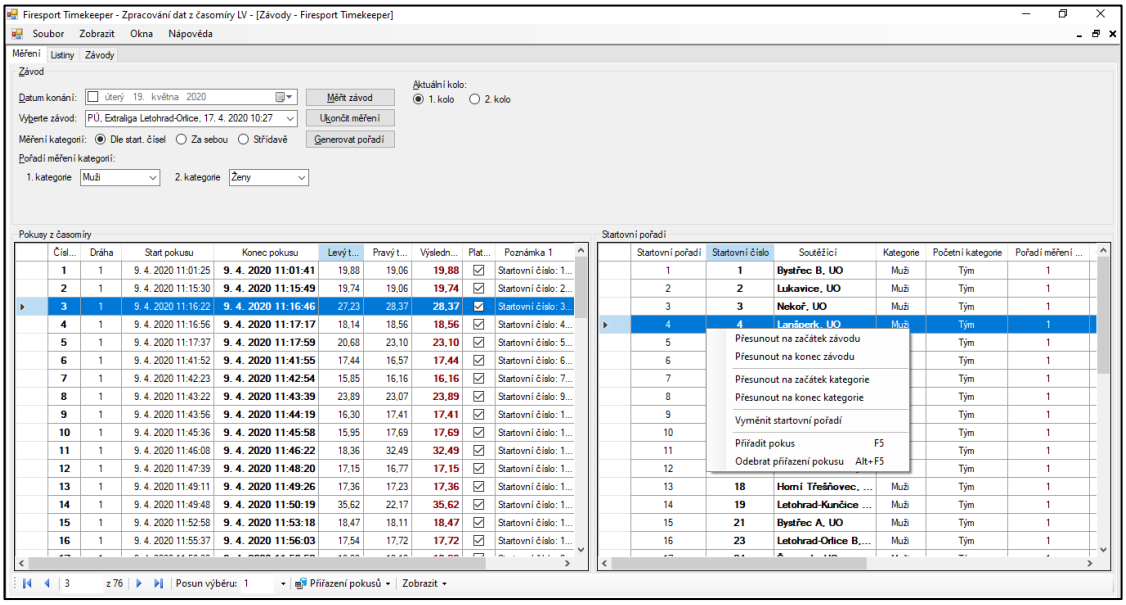
Grafický návrh modulu Závody

Modul Závody již spolupracuje s databází. S jeho pomocí do ní můžeme vkládat nové závody, nebo upravovat či mazat ty stávající. Miniprogram umožňuje také vytvářet startovní a výsledkové listiny soutěže. Ke vzniklým startovním pořadím pak jednoduše přiřazujeme pokusy naměřené modulem Časomíra.

V tomto modulu můžeme pracovat s velkým množstvím dat, která jsou soustředěna do tabulek. Je zde tak větší riziko chyby zaviněné např. nepozorností obsluhy časomíry. V GUI tak musí být kladen ještě větší důraz na důležité údaje obsažené v některých buňkách tabulky.

Stěžejní hodnoty jsou proto v tabulkách vyznačeny např. tučným písmem, jinou barvou atd., aby upoutaly pozornost lidského oka. Grafické rozhraní je dále navrženo tak, aby v případě chybné obsluhy šlo, pokud možno, zjednat nápravu.

V GUI modulu Závody je základním ovládacím prvkem *TabControl*. Formulář je díky němu rozdělen do více karet, přičemž každá tato karta obsahuje odlišné ovládací prvky sloužící k předání informací uživateli. Ve zmíněném hlavním ovládacím prvku nalezneme 3 karty. Každá z nich, ač obsahuje jiné údaje, je rozdělena pohyblivým panelem *SplitContainer* do základních tří částí. V té první se nachází řídící prvky sloužící k filtrování zobrazovaných dat v tabulkách. Tyto vybrané záznamy jsou pak obsluze programu zobrazeny pomocí *DataGridView*, jehož vzhled se velmi podobá známému tabulkovému rozhraní z Microsoft Excelu. Uvedený ovládací prvek je optimalizovaný pro práci s datovými zdroji. Data v něm můžeme řadit (umožňuje-li to i programová část aplikace), taktéž je možné měnit pořadí jednotlivých sloupců tabulky. Poslední částí je panel nástrojů, na kterém se nachází ovládací součásti pro navigaci mezi záznamy tabulky, k úpravě dat a také prvky pro různé možnosti

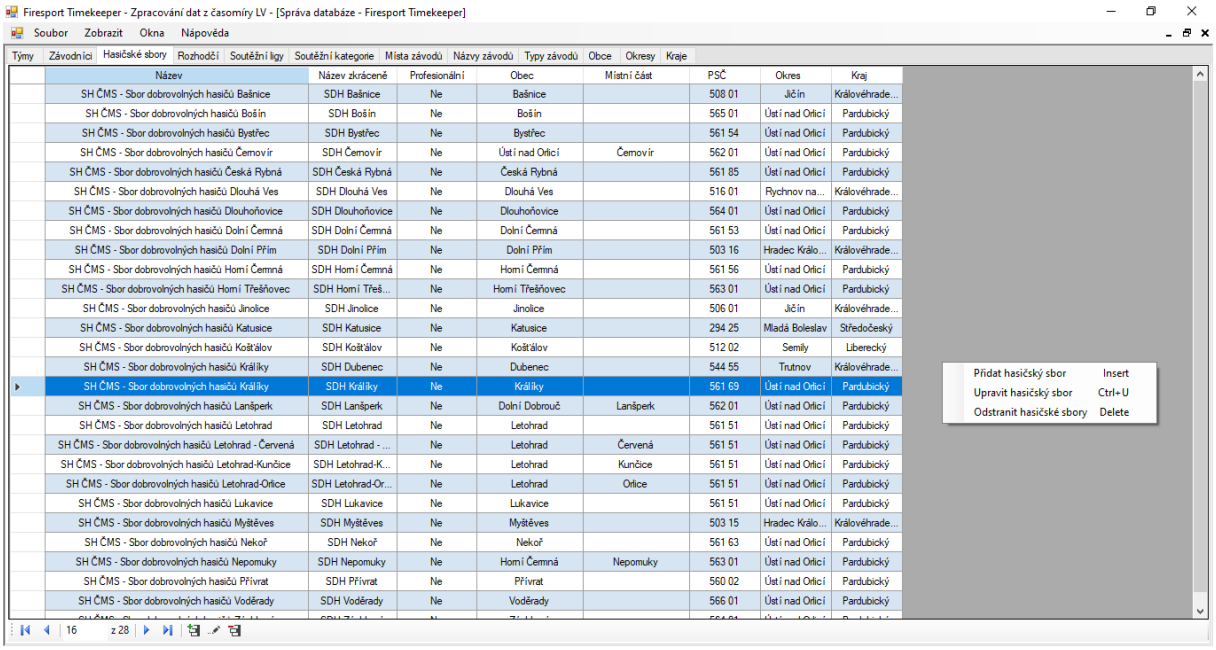


Obrázek 17 – GUI modulu Závody

zobrazení, případně exportu dat do souboru. Při kliknutí pravým tlačítkem na některé tabulky je uživateli zobrazena místní nabídka s možnostmi.

Grafický návrh modulu Správa databáze

Tento modul je určen ke správě dat tabulek v databázi. Design grafického uživatelského rozhraní se velmi podobá designu GUI modulu Závody (viz Grafický návrh modulu Závody). Také je zde použit *TabControl*. Jeho karty odpovídají konkrétním tabulkám v databázi. Na těchto kartách se jednotlivě nachází *DataGridView* zobrazující data uživateli a také stavový řádek pro navigaci a úpravu těchto údajů. Ovládací prvky pro filtrování dat zde použity nejsou. Tabulky obsahují i místní nabídku, jež se při kliknutí pravým tlačítkem myši zobrazí.

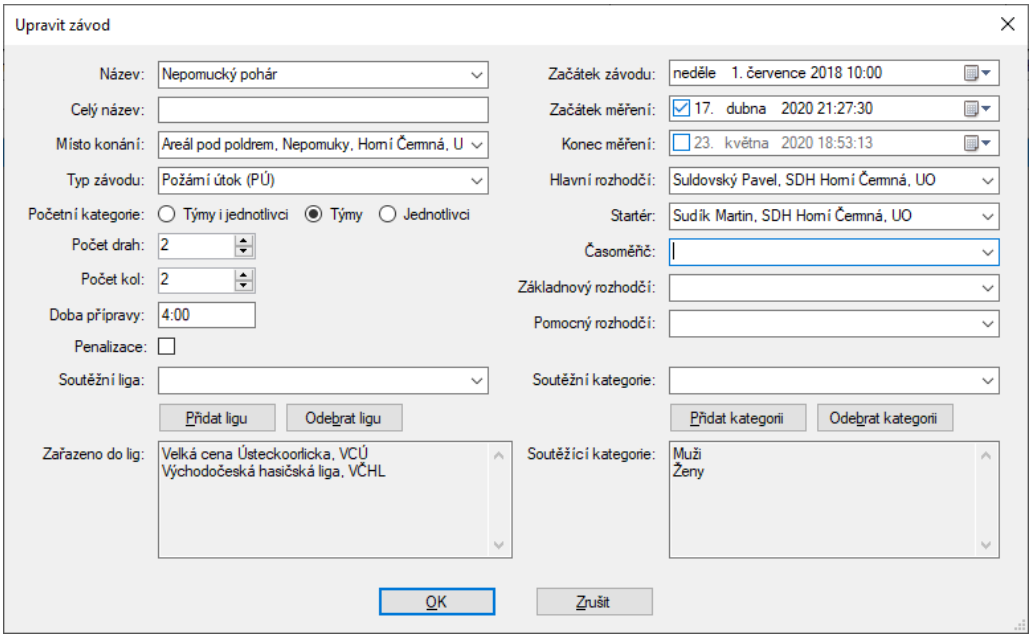


Obrázek 18 – GUI modulu Správa databáze

Dialogová okna pro vkládání a editaci dat

Pro úpravu a vkládání dat tabulek uživatelem byla použita vlastní dialogová okna. Jedná se o formuláře Windows Forms, jež obsahují specifické ovládací prvky pro konkrétní měněné údaje. Přestože některé řídící elementy, jako např. *DataGridView*, umožňují editaci přímo v buňkách grafického rozhraní tabulky, je způsob úprav prostřednictvím dialogového okna preferován především z důvodu větší přehlednosti při vkládaných a upravovaných údajích. Při ztížených pracovních podmínkách, např. v případě zobrazení aplikace na monitorech s nižším rozlišením, je navíc větší riziko vzniku chyby. Zejména u tabulek *DataGridView* s větším množstvím sloupců budou totiž zobrazeny posuvníky, které přispívají k nepřehlednosti zobrazených dat. Když k tomu připočteme ještě obvyklou situaci na závodech, kde je míra

soustředěnosti obsluhy programu značně omezena okolními vlivy (hluk, ztížené pracovní podmínky…), je problém na světě. Cílem návrhu GUI však bylo těmto problémům zabránit.



Obrázek 19 – GUI dialogového okna pro editaci údajů o závodu

5.2.2 Ovládání aplikace

K řízení aplikace slouží ovládací prvky. Řeč o nich byla již v 5.2.1. V reakci na události těchto prvků (najetí kurzorem myši, kliknutí…) můžeme měnit chování programu.

Všechny ovládací prvky použité pro implementaci programu jsou součástí knihovny Windows Forms. Není tak nutné programovat jejich grafickou a funkční část, ale stačí vytvořit pouze obslužnou metodu v reakci na existující událost, jejíž mechanismus je taktéž předprogramován.

Uživatelské rozhraní ovládacích prvků je optimalizováno především pro obsluhu kurzorem myši. Na to bylo při vývoji aplikace myšleno. Výhodou použitého způsobu ovládání je zejména možnost zobrazení poznámek při přesunu kurzoru myši nad daný ovládací prvek. Toho je v aplikaci mnohokrát využito. Při „najetí myši“ na prvek, jenž v sobě „skrývá další sdělení,“ je uživatel obvykle upozorněn změnou podoby kurzoru. Po chvíli se zobrazí i text této poznámky. Ta v sobě nejčastěji nese informace v podobě rad a tipů využitelných při používání programu. Aplikaci tak může obsluhovat i méně znalý uživatel.

Obsluhu některých ovládacích prvků lze provádět i prostřednictvím klávesových zkratek. Aplikace této možnosti taktéž využívá, ale pouze jako sekundárního způsobu ovládání. Příslušné klávesové zkratky jsou ve většině případů uvedeny v popisku konkrétního ovládacího prvku, u nějž je druhotní možnost obsluhy dostupná.

5.3 Relační databáze pro ukládání výsledků soutěží

Miniaplikace Časomíra umožňuje export naměřených pokusů s vlastními poznámkami do formátu CSV bez použití databáze. Budeme-li však chtít výsledky zároveň vyhodnotit, musíme využít dalších dvou miniprogramů, jež komunikují s DB. V té jsou uloženy především informace o soutěžících, závodech a startovních listinách těchto klání. Jsou-li ke vzniklým startovním pořadím přiřazeny i naměřené časy, je možné s pomocí DB jednoduše vytvořit také listiny výsledkové. Pojďme si nyní databázi představit podrobněji.

Pro ukládání a zpracování naměřených výsledků z časomíry byla relační databáze použita především pro možnost centrálního ukládání a zpracování uložených dat. Tyto údaje je pak možné sdílet mezi více aplikacemi, což je do budoucna plánováno. DB je navržena pro systém Microsoft SQL Server 2019. Její projekt byl implementován v programu Microsoft SQL Server Management Studio 18. Samotná databáze nesoucí název *CasomiraDB* se skládá z tabulek a relací mezi nimi, triggerů, funkcí a uložených procedur.

5.3.1 Databázové tabulky

Tyto tabulky jsou základními objekty každé relační databáze. Slouží k organizovanému uložení dat do paměti DB. Aby to bylo možné, musí být tyto položky předem definovaného datového typu. Nelze tedy uložit „cokoliv kamkoliv.“ Pokud použijeme asociaci s klasickou dvourozměrnou tabulkou, názvy sloupců nám říkají, jak mají data vypadat (jsou definované datové typy) a jednotlivé řádky jsou pak uložená data. Každý řádek v tabulce musí mít přesný počet definovatelných datových typů tabulky – všechny řádky dané tabulky tedy mají stejný počet sloupců. [22]

Při návrhu tabulek byl kladen důraz především na zamezení redundance (nadbytečnosti dat) a na unikátnost dat v databázi. Může totiž nastat situace, že některé hodnoty datových typů budou ve více řádcích obsahovat stejný údaj. Uložení stejné informace opakovaně do každého řádku je však velmi nepraktické především při úpravě těchto hodnot, kdy musíme údaj změnit u všech výskytů v databázi. Řešením je uložení duplicitních dat do samostatné tabulky a použití databázových relací. Při návrhu tabulek bylo taktéž myšleno na možnost dalšího využití v budoucnu. Databáze proto počítá i s využitím např. pro webovou aplikaci, v níž by bylo možné online zobrazit výsledky či různé statistické údaje závodů, soutěžících, případně soutěžních lig.

Databázové relace

Pojem „databázová relace“ značí vztah mezi dvěma, případně více tabulkami. Rozlišujeme několik druhů relací:

- **Bez relace** – Žádný vztah mezi tabulkami
- **1:N** – 1 záznam v tabulce A odpovídá několika záznamům v tabulce B.
- **M:**N – Několik záznamů v tabulce A odpovídá několika záznamům v tabulce B.
- **1:1** – 1 záznam v tabulce A odpovídá 1 záznamu v tabulce B.

K vytvoření této vazby je zapotřebí jednoznačně definovat záznam (řádek) v tabulkách. K tomuto účelu se často do tabulek přidává definovaný datový typ „ID.“ Jedná se o unikátní záznam řádku tabulky (tzv. primární klíč), který je obvykle automaticky generován. V některých tabulkách je také dbáno na entitní integritu. To znamená, že do takové tabulky nelze přidat řádek se stejnými hodnotovými daty vícekrát. To se může hodit např. v tabulce okresů či krajů ČR (neexistuje více krajů se stejným názvem). [21]

Databáze *CasomiraDB* je poměrně rozsáhlá. Obsahuje 18 tabulek. Z toho jsou tři pomocné, neboť DB neumožňuje přímé vytvoření relace M:N. Problém se řeší právě vytvořením pomocné, tzv. spojové, tabulky a použití dvou relací 1:N mezi tabulkou A, spojovou tabulkou a mezi tabulkou B a opět spojovou tabulkou. Všechny tabulky v databázi obsahují jednoduchý popis, v němž jsou uvedeny informace o ukládaných datech. Také definované datové typy všech tabulek obsahují stručnou charakteristiku ukládané hodnoty tohoto parametru záznamu. Představme si nyní ty vůbec nejdůležitější tabulky, jež jsou v DB implementovány.

Prvním takovým případem je tabulka *Zavody*. V té jsou uloženy všechny měřené závody s podrobnými údaji o těchto kláních. Kromě názvů soutěží, lokací a datumů konání to jsou hlavně informace o parametrech a průběhu měření – soutěžní kategorie, početní kategorie a soutěžní ligy, do nichž je soutěž zařazena, časy zahájení a ukončení samotného měření a další informace, jež je možné získat z propozic daných závodů.

Druhá tabulka v pořadí s názvem *Pokusy* obsahuje údaje o všech zaznamenaných pokusech daných soutěžících. Nejdůležitějším definovaným údajem je samozřejmě naměřený výsledný čas, a pak také informace o platnosti pokusu. Následuje identifikační číslo startovního pořadí, ke kterému byl pokus přiřazen. Další údaje jsou již doplňující – jedná se o přesné časy startů a konců příprav k provedení pokusu a také samotných měření sportovních výkonů, případně různé poznámky k jednotlivým záznamům.

Třetí důležitá tabulka nese název *Startovni_poradi*. Ta se relačně odkazuje na další tabulky – již zmíněné *Zavody* i *Pokusy* a dále na tabulky s informacemi o soutěžících. V praxi to tedy znamená, že je závodníkovi, jenž se účastní v dané soutěži, přiděleno unikátní startovní pořadí s informací o startovním čísle v závodu. Po provedení pokusu je ke vzniklému pořadí ještě přidělena reference na naměřený výsledek z tabulky pokusů.

V dalších tabulkách jsou uloženy různé informace o soutěžících, rozhodčích, hasičských sborech, názvech, místech a druzích závodů, nebo soutěžních kategoriích. Některé tabulky jsou mezi sebou relačně provázány.

Relace jsou pak v DB definovány pomocí tzv. cizího klíče. Ten nám kontroluje, zda existuje hodnota přidávaná do vybraného sloupce referenční tabulky v určeném sloupci tabulky zdrojové. Reálná data jsou tudíž uložena pouze na 1 místě, zatímco ostatním tabulkám je poskytován pouze odkaz na originální údaje. Cizí klíče taktéž umožňují definovat akce při odstranění záznamu ze zdrojové tabulky, na nějž se některé sloupce referenčních tabulek stále odkazují. [22]

Všechny databázové tabulky a relace jsou podrobně znázorněny v tzv. ER diagramu, který je přílohou této bakalářské práce. Definice sloupců databázových tabulek, reprezentujících definované datové typy jednotlivých položek záznamu, jsou taktéž součástí dodatku bakalářské práce.

5.3.2 Triggery

Trigger si můžeme představit jako souhrn činností, které se mají provést při definované změně databázové tabulky. Ve SQL databázi existuje obvykle 6 těchto definovaných spouštěcích událostí – před a po vložení nových řádků, úpravě stávajících dat či smazání existujících záznamů z dané tabulky. Při práci s triggery můžeme pracovat s obrazy vázané tabulky před provedením a po provedení akce triggeru. Pomocí těchto spouštěčů tak můžeme např. zabránit smazání dat za určitých podmínek. Je reálné také provést dodatečnou úpravu vkládaných či upravovaných záznamů, a to dokonce i v jiné tabulce, než té, na kterou se trigger váže. [22]

Databáze *CasomiraDB* obsahuje celkem 23 triggerů definovaných na různých tabulkách. Tyto spouštěče slouží především ke kontrole vkládaných a upravovaných startovních pořadí s informacemi o soutěžících v daných kláních. Parametrům závodu uvedených ve stejnojmenné tabulce totiž nemusí vyhovovat všichni soutěžící. Pokud zmíněný případ nastane a uživatel se pokusí přidělit tomuto závodníkovi startovní pořadí do odpovídající tabulky, trigger vyvolá chybu a vrátí podobu tabulky do stavu před její nepovolenou změnou. Stejně tak pokud je ke

startovnímu pořadí v konkrétním závodě přidělen naměřený pokus s nevyhovujícími parametry (např. jedná se o pokus požárního útoku, zatímco v parametrech závodu v DB je jako typ soutěže uvedena Štafeta 4 x 100 metrů s překážkami), je triggerem vyvolána chyba a k uložení dat do databáze nedojde. Tyto potencionální chyby je nutné kontrolovat i při změně parametrů závodu či u editace soutěžících, jež už mají přidělena nějaká startovní pořadí. Proto je spouštěčů poměrně dost, neboť jsou, ač mnohdy se stejným souhrnem obsluhujících příkazů, definovány nad více tabulkami.

V databázi jsou dále použity triggery pro kontrolu duplicit startovních pořadí přidělených 1 soutěžícímu vícekrát v závodě, nebo pro hlídání přiřazení povolených soutěžních lig či kategorií soutěžících. Každý implementovaný trigger je v databázi opět popsán, aby bylo rychle zřejmé, k čemu slouží.

5.3.3 Databázové funkce

Krom vestavěných databázových funkcí umožňuje MSSQL vytváření vlastních funkcí. Ty mohou být různých typů, od tabulkových, vracejících data v tabulce, přes agregační, umožňující seskupování dat na základě definovaných pravidel, po nejjednodušší – skalární funkce, které obvykle pracují pouze s jedním výstupním argumentem. [22]

V navrhované databázi jsou deklarovány 3 skalární funkce. První slouží pro vrácení údajů o rozhodčích ze stejnojmenné tabulky v textové podobě a další 2 se týkají GPS souřadnic. Těch je využito v tabulce *Mista_zavodu*. Zmíněné souřadnice se ukládají v databázovém formátu „geography.“ MSSQL ale neobsahuje funkce pro převod na tento typ ze 2 reálných čísel reprezentujících zeměpisnou šířku a délku, a také chybí funkce pro převod na textový formát WGS84, jenž se pro GPS souřadnice standardně používá. Zmíněné nedostatky uvedené dvě funkce kompenzují.

5.3.4 Uložené procedury

Uloženou procedurou rozumíme části SQL příkazů, jež můžeme využít opakovaně pro práci s daty dané databáze. Stejně jako u funkce, můžeme i v proceduře definovat vstupní a výstupní parametry. Velkou výhodou je pak podpora využití uložených procedur v běžně používaných programovacích jazycích. Tato přednost umožňuje sdílet kód procedur pro práci s databázemi i mezi více aplikacemi. Mezi nevýhody deklarace uložených procedur v databázi pak patří především menší přehlednost kódu a omezené možnosti jazyka SQL. Při implementaci stejné procedury např. v C# s využitím LINQ příkazů a následným použitím technologie

LINQ to SQL by tedy bylo možné s databází pracovat plně objektově, což je výhodou pro programátora. Objektově orientované programování totiž obecně umožňuje rychlejší vývoj a snadnější úpravu či údržbu kódu vyvíjených aplikací. [22]

Uložené procedury představují největší část objektů navrhované databáze. Ve vyvíjené DB je jich deklarováno 121. Uvědomíme-li si však fakt, že pro základní správu jedné tabulky je potřeba až 5 procedur a že databáze obsahuje 18 tabulek, není číslo nikterak vysoké.

Při návrhu uložených procedur databáze bylo počítáno s možností využití těchto konání pro další aplikace. Části kódu, které zajišťují práci s daty v DB, tak nejsou implementovány v jednotlivých aplikacích, ale už v samotné databázi prostřednictvím zmíněných procedur. Těch pak programy pracující s DB mohou využít.

Jak již bylo řečeno, velká část procedur slouží pro správu databázových tabulek. K základním operacím údržby tabulkových dat bylo potřeba definovat procedury pro vložení, úpravu a mazání dat, pro výpis celé tabulky i vybraného záznamu tabulky jednotlivě.

Další procedury byly použity především pro vytváření startovních a výsledkových listin dle zadaných vstupních parametrů, pro přesuny a záměny startovních pořadí vybraných závodníků, a především pro ukládání a přiřazování pokusů z časomíry ke startovním pořadím v databázi. Pro ukázku si zde jednu deklaraci uložené procedury představíme:

```
USE [CasomiraDB]
GO
/****** Object:  StoredProcedure [dbo].[UlozPokus]    Script Date: 22.05.2020 11:01:11 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[UlozPokus] @StartovniPoradiID int = NULL, @StartPripravy datetime2(7) = NULL, @KonecPripravy datetime2(7) = NULL, @StartPokusu datetime2(7) = NULL,
                        @KonecPokusu datetime2(7) = NULL, @CasLTercSekundy float = NULL, @CasPTercSekundy float = NULL, @CasDraha float = NULL, @Penalizace float = NULL,
                        @Platny bit = 0, @PocetOpakovani int = 0, @CisloDrahy int = NULL, @CisloMereni int = NULL, @RucniMereni bit = 1,
                        @Poznamka1 nvarchar(max) = NULL,  @Poznamka2 nvarchar(max) = NULL
AS
BEGIN TRANSACTION
IF @StartovniPoradiID IS NULL OR NOT EXISTS(SELECT * FROM Startovni_poradi WHERE Startovni_poradi.ID = @StartovniPoradiID)
BEGIN
ROLLBACK TRANSACTION
RETURN 0
END

IF EXISTS(SELECT * FROM Pokusy
          INNER JOIN Startovni_poradi ON  Pokusy.Startovni_poradi_ID = Startovni_poradi.ID
          INNER JOIN Zavody ON Startovni_poradi.Zavod_ID = Zavody.ID
          WHERE Startovni_poradi_ID = @StartovniPoradiID AND Pokusy.Cislo_kola = Zavody.Aktualni_kolo)
BEGIN
UPDATE Pokusy
SET
Start_pripravy = @StartPripravy, Konec_pripravy = @KonecPripravy, Start_pokusu = @StartPokusu, Konec_pokusu = @KonecPokusu,
Cas_levy_terc = @CasLTercSekundy, Cas_pravy_terc = @CasPTercSekundy, Cas_draha = @CasDraha, Penalizace = @Penalizace,
Platny = @Platny, Pocet_opakovani = @PocetOpakovani, Cislo_drahy = @CisloDrahy, Cislo_mereni = @CisloMereni,
Rucni_mereni = @RucniMereni, Poznamka_1 = @Poznamka1, Poznamka_2 = @Poznamka2

FROM        (SELECT Startovni_poradi.ID, Zavody.ID AS Zavod_ID, Zavody.Aktualni_kolo
             FROM Startovni_poradi
             INNER JOIN Zavody ON Startovni_poradi.Zavod_ID = Zavody.ID
             WHERE Startovni_poradi.ID = @StartovniPoradiID) AS dotaz

WHERE       Startovni_poradi_ID = @StartovniPoradiID AND Cislo_kola = dotaz.Aktualni_kolo
END
ELSE BEGIN
INSERT INTO Pokusy
           ([Startovni_poradi_ID], [Start_pripravy], [Konec_pripravy], [Start_pokusu], [Konec_pokusu],
           [Cas_levy_terc], [Cas_pravy_terc], [Cas_draha], [Penalizace], [Platny], [Pocet_opakovani],
           [Cislo_kola], [Cislo_drahy], [Cislo_mereni], [Rucni_mereni], [Poznamka_1], [Poznamka_2])

SELECT      @StartovniPoradiID, @StartPripravy, @KonecPripravy, @StartPokusu, @KonecPokusu,
            @CasLTercSekundy, @CasPTercSekundy, @CasDraha, @Penalizace, @Platny, @PocetOpakovani,
            Zavody.Aktualni_kolo, @CisloDrahy, @CisloMereni, @RucniMereni, @Poznamka1, @Poznamka2
FROM        Startovni_poradi
            INNER JOIN Zavody ON Startovni_poradi.Zavod_ID = Zavody.ID
            WHERE Startovni_poradi.ID = @StartovniPoradiID
END
COMMIT TRANSACTION
RETURN 1
```

Obrázek 20 – Ukázka SQL kódu uložené procedury

V první části SQL kódu je uvedena deklarace samotné procedury – jejího názvu a vstupních parametrů s výchozími hodnotami. Následně se v transakci (transakce zajišťuje provedení více příkazů naráz a v případě nějaké chyby odvolá změny vykonané již dříve provedenými příkazy transakce) přistupuje k větvení kódu. Pokud neexistuje záznam v tabulce startovních pořadí identifikovaný vstupním parametrem *@StartovniPoradiID,* je transakce spolu s procedurou ukončena. V opačném případě se dotazujeme, jestli už byl k identifikovanému startovnímu pořadí v aktuálním soutěžním kole závodu přiřazen nějaký pokus. Pokud ano, údaje o sportovním výkonu se aktualizují (není dovoleno 1 soutěžícímu v jednom soutěžním kole závodit víckrát). V opačném případě dojde k vložení nového řádku s údaji předanými vybranými parametry procedury do tabulky pokusů. Nakonec je transakce potvrzena a procedura vrátí hodnotu „1“ reprezentující její úspěšné provedení.

5.4 Moduly aplikace Firesport Timekeeper

Zatímco v 5.2.1 byla řeč o vzhledu programu, v tomto oddíle si detailněji probereme funkční stranu aplikace. Podrobně si popíšeme, k čemu jednotlivé moduly slouží a jak fungují. Představíme si nejzákladnější objekty, bez nichž by program nemohl fungovat a dojde i na ukázku a následný popis částí kódu programu.

5.4.1 Modul Časomíra

Jak již bylo naznačeno v 5.2, tento modul slouží především ke zpracování dat z časomíry a k ukládání naměřených časů do paměti nebo souboru. Takto uložené časy mohou být využity k dalšímu zpracování.

Nejprve bylo nutné vytvořit reálný model světa. Pro základní obraz této skutečnosti slouží tři třídy s názvy *Pokus*, *Mereni* a *Casomira.*

Třída *Pokus* definuje vlastnosti skutečného pokusu, jenž může být časomírou naměřen. Jedná se o abstraktní třídu. To znamená, že ji nelze použít pro definici konkrétního objektu (v tomto případě pokusu), ale je nejprve nutné od této třídy odvodit potomka. Ti v projektu existují 2 – třída *PokusUtok,* sloužící pro uložení pokusu naměřeného v režimu časomíry pro měření PÚ a *PokusStafeta,* která je určena pro ukládání časů měřených v režimu „štafeta.“ Třídy obsahují krom informací o pokusu a metod sloužících pro práci s objekty též definici události vyvolané v případě, že dojde ke změně některého z údajů o pokusu.

Analogicky s nastaveným počtem drah a typem měření skutečné časomíry pak třída *Mereni* zapouzdřuje skupiny pokusů. Bude-li tedy na časomíře nastaveno měření požárního

útoku pro 2 dráhy, odpovídající instance třídy *Mereni* bude obsahovat pole 2 pokusů typu *PokusUtok*. Dále jsou zde implementovány metody společné pro všechny zapouzdřené pokusy – např. pro start přípravy k pokusu, pro společný start nebo ukončení měření pokusů apod.

Třída *Casomira* pak reprezentuje vlastnosti skutečné časomíry. Zapouzdřuje jednotlivá měření skupin pokusů, které se ukládají do kolekce seznamu reprezentovaného generickou (pro různé datové typy) třídou *List,* dále umožňuje změnit nastavení měření analogicky s hlavní jednotkou časomíry LV. *Casomira* dále obsahuje metody, které se podobají konkrétním schopnostem reálné časomíry a slouží pro práci s datovými typy této třídy. Jedná se o metody pro zpracování hexadecimálních dat ze sériové linky nebo pro uložení/načtení dat do/ze souboru. Ve třídě je dále deklarováno několik událostí sloužících k informování jiných objektů, že nastala nějaká situace. Jmenovitě jsou to události informující o spuštění přípravy k pokusu, zahájení i ukončení měření pokusu, změně nastavení časomíry nebo počtu všech měření, o úpravě údajů uživatelem po naměření pokusů, či o zobrazení posledních časů, o připravenosti cílových zařízení k dalšímu pokusu a v neposlední řadě o vynulování časomíry.

Sériová komunikace

Pro sériovou komunikaci mezi programem a časomírou byla využita třída *SerialPort,* jež je součástí .NET Frameworku. Zpracování dat je pak vyřešeno implementací ošetřující metody na deklarovanou událost *DataReceived,* která je vyvolána po přijetí dat z komunikujícího zařízení. K jejich čtení je použita metoda *Read* zmíněné třídy, jež jednotlivé bajty uloží do pole. Problém však nastává při očekávání vícebajtové zprávy. Data totiž nemusí přijít najednou.

Takovým příkladem je i příjem výsledných časů z časomíry. Ta totiž nejdříve odešle příznakový bajt informující o tom, že bude posílán výsledný čas, a následně zašle čtyřbajtovou informaci s tímto časem a číslem dráhy, na níž byla hodnota naměřena. Vše je řešeno pomocí metody *ZpracujBajty* v třídě *Casomira*. Pokud je takový příznak indikován, metoda zkontroluje celkovou délku přijaté zprávy. Je-li zjištěno, že informace nepřišla naráz, bajty zůstanou

```
//5 bajtová položka
else if (bufferSerial[i] == 0xBA)
{
    if (bufferSerial.Count - i >= 5)     //pro rozkódování zpráva přišla celá
    {
        List<byte> bajtyCasu = new List<byte>();
        for (int j = 0; j < 5; j++)
        {
            bajtyCasu.Add(bufferSerial[i + j]);     //překopíruje 5 bajtů času do nového Listu
        }
        NastavVysledkyCas(bajtyCasu);               //metoda řešící zpracování času

        i += 4;     //nakonec inkrementuju i
    }
    else            //pro rozkódování zpráva nepřišla celá
    {
        bufferSerial.RemoveRange(0, i);         //odebere již zpracované bajty
        return;
    }
}
```

Obrázek 21 – Ukázka C# kódu – Zpracování dat z časomíry

uloženy v bufferu implementovaným vlastní třídou a budou zpracovávány až při dalším přijetí dostatečného množství dat.

Průběh měření s využitím modulu

Po spuštění modulu je nutné propojit hlavní jednotku časomíry a PC pomocí sériového kabelu. Neobsahuje-li počítač vstup pro sériový port, je možné použít převodník USB-232. Poté můžeme spustit program. Následně se objeví dialogové okno pro připojení k sériové lince. Vybereme příslušný port a tlačítkem „Připojit“ zahájíme komunikaci PC. Následně připojíme časomíru k elektrickému zdroji. Po tomto kroku časomíra zašle přes sériovou linku hodnoty nastavení. Pokud byla časomíra již zapnuta, je nutné ji pro inicializaci vypnout a zapnout, nebo vyvolat nastavení časomíry, po jehož ukončení se bajty do PC opět zašlou. Následně by mělo dojít k inicializaci programu. Poté můžeme spustit přípravu k pokusu.

Nastane-li tato situace, do PC je odeslán příznak spuštění přípravy. Po jeho detekci programem je zaznamenán aktuální čas počítače reprezentující přesný čas zahájení přípravy a spustí se odpočet. Dojde-li k uplynutí nastavené doby nebo k odstartování pokusu v průběhu odpočtu, časomíra opět odešle příznak reprezentující příslušnou událost. V PC je znovu zaznamenán aktuální čas, který reprezentuje čas ukončení přípravy a v případě startu měření zároveň čas startu pokusu. Může taktéž dojít k vynulování časomíry, které zruší aktuální měření. Pro zjednodušení počítejme s tím, že byl pokus odstartován před uplynutím doby přípravy. Přesto nyní může nastat více situací.

U prvního případu je časomíra vynulována v časovém limitu od spuštění měření, definovaném v nastavení tohoto modulu, a nebyl naměřen žádný výsledný čas. Tehdy je vynulování časomíry vyhodnoceno jako předčasný start a pokus bude opakován. Není-li tomu tak, můžeme vyhodnocení programu ručně změnit na typ druhý.

V druhé situaci je vynulování provedeno po stanoveném limitu, nebo sice ve vymezeném čase, ale bylo již sepnuto některé cílové zařízení. Tento případ program vyhodnotí jako neplatný pokus. Aplikace opět zaznamená aktuální čas, který reprezentuje dobu ukončení pokusů na všech drahách s nesepnutými cílovými prvky. Poté se přechází k dalšímu měření. Vyhodnocení programu je možné ručně změnit na první situaci.

Ve třetím případě k vynulování časomíry nedojde a všechny cílové prvky sepnou, čímž dojde k ukončení měření pokusů všech drah. Zde je vždy po sepnutí cílového prvku zaslána do PC informace o této skutečnosti spolu s výsledným časem a číslem dráhy. Počítač opět zaznamená aktuální čas, který je v případě sepnutí všech cílových prvků na dráze rovněž časem ukončení pokusu na této dráze.

Následně je obsluhou časomíra vynulována a je možné přistoupit k dalšímu měření. Při vynulované časomíře je rovněž možné obsluhou hlavní jednotky měřícího zařízení zobrazit poslední naměřené časy nebo změnit nastavení měření. O těchto případech je opět PC přes sériovou linku informováno. Kód metody *VynulujMereni*, jež je součástí třídy *Casomira* a odráží skutečné chování reálné časomíry při vynulování, je zobrazen v následující ukázce.

```
Počet odkazů: 1
public void VynulujMereni()
{
    //Metoda zkontroluje, zda byl nastaven typ měření. Pokud ne, vyvolá výjimku.
    ZkontrolujMereni();

    if (ZobrazenyPosledniCasy)          //Uživatel pouze zobrazil na časomíře časy, teď to ruší a vrací se do zákl. obrazovky
    {
        ZobrazenyPosledniCasy = false;
        OnPosledniCasyZobrazeny(EventArgs.Empty);
        return;
    }

    if (AktualniMereni.JeVynulovano() || PocetDrah == 0) return;         //Již vynulováno, nebo není co nulovat

    if (AktualniMereni[1].DejStatutPokusu() == Udalost.Nezahajeno)        //Pokud běžela příprava (stačí u 1 dráhy, když měření přípravy se spouští stejně)
    {
        bool ponechatPripravu = AktualniMereni.JePlatnaUdalostPriprava(Udalost.Ukonceno) ? true : false;       //Ponechám čas přípravy, pokud uběhne celá....
        AktualniMereni.VynulujPokusy(ponechatPripravu || Nastaveni.NenulovatCasyPripravy);
        if (Nastaveni.NenulovatCasyPripravy) AktualniMereni.UkonciPripravu(DateTime.Now);
        OnCasomiraVynulovana(new CasomiraVynulovanaEventArgs(TypNulovani.PripravaUkoncena));
        return;
    }

    if (ZjistiDalsiMereni(out DateTime nyni))               //Zjistí, zda se jedná o ulitý/nesprávně provedený start, či o NP
    {
        bool ukoncit = false;
        for (int i = 1; i <= AktualniMereni.PocetDrah; i++)      //Doplnění čísel drah a časů konců měření pro pokusy
        {
            if (AktualniMereni[i].DejStatutPokusu() == Udalost.Probihajici)
            {
                DokonciProbihajiciPokus(i, nyni);
                ukoncit = true;
            }
        }
        if (ukoncit)            //Měření ukončeno, neplatí pro předč. start a vynulování!
        {
            UkonciMereni(false);
        }

        //Přidá nové měření
        PridejMereni(TypMereni, PocetDrah, DobaPripravy, true);

        OnCasomiraVynulovana(new CasomiraVynulovanaEventArgs(ukoncit ? TypNulovani.NeplatnyPokus : TypNulovani.None));
    }
    else
    {
        casPosledniPredcasnyStart = AktualniMereni[1].StartPokusu;        //Potřebné hlavně pro zpětné udělení NP při vyhodnocení programem jako předč. start

        AktualniMereni.VynulujPokusy(true);                       //opakovaný start, true ponechá časy startu přípravy
        OnCasomiraVynulovana(new CasomiraVynulovanaEventArgs(TypNulovani.PredcasnyStart));
    }
}
```

Obrázek 22 – Ukázka C# kódu – Vynulování měření

Volaná metoda z ukázky s názvem *ZjistiDalsiMereni* zaznamená aktuální čas PC a zjišťuje, zda byly všechny pokusy ukončeny. Pokud ne, vypočte dobu od startu odečtením aktuálního času od času startu daného pokusu. Pokud je tento interval delší, než je nastavená maximální doba od startu pokusu, vrací metoda hodnotu *true*. Jinak je výsledkem nepravda.

V případě, že je zjištěna skutečnost dalšího měření, jsou aktuální pokusy ukončeny s nynějším časem předaným ze zjišťující metody. Rovněž je k těmto pokusům doplněno číslo dráhy. To však není možné jednoduše zjistit u typu měření „požární útok“ na jednu dráhu, kdy reálná časomíra umožňuje sepnutí časů na drahách obou To je rozdíl oproti měření v módu „štafeta,“ kdy při nastavení např. 3 drah spínají vždy pouze dráhy s čísly 1-3 dle označení na rozbočovači pro cílová zařízení. Číslo dráhy je tak znatelné až po sepnutí některého z cílových prvků (terčů), kdy je informace o dráze zaslána spolu s výsledným časem do PC a na této dráze

je očekáváno taktéž sepnutí druhého koncového zařízení. Pokud však k takovému sepnutí nedojde a jedná se o NP, je nutné dráhu obsluhou časomíry doplnit. V opačném případě bude na stavovém řádku zobrazeno varování s tímto nedostatkem a nebude umožněno přiřazení takového pokusu do databáze.

V ukázce můžeme také vidět využití metod sloužících k vyvolání příslušných událostí třídy *Casomira*. Názvy těchto metod začínají předponou „On“ a jsou volány vždy po provedení příslušných kroků, jež zapříčiní změnu určité skutečnosti. V argumentech těchto metod jsou taktéž předávány doplňující informace o nastalé změně, které budou metodám jiných objektů registrovaných v události rovněž předány.

5.4.2 Modul Závody

Z kapitoly 5.2 již víme, že modul Závody je jakýmsi prostředníkem mezi modulem Časomíra, ze kterého tento zprostředkovatel přebírá naměřené časy, a databází. Do ní modul Závody tyto časy ukládá. Stává se tak po přiřazení výsledných hodnot z časomíry jejich „majitelům“ – soutěžícím, kterým naměřený výkon patří.

Stěžejním bodem pro implementaci modulu Závody bylo zprovoznění komunikace aplikace s databází. K této problematice bylo využito technologie ADO.NET, jež je opět součástí .NET Frameworku. Jedná se o sadu tříd, jejichž služby umožňují přístup k datům a tvorbu databázových aplikací. ADO.NET disponuje více možnostmi pro komunikaci s databází.

V dalších krocích bylo nutné databázová data zpracovat a zobrazit je uživateli. Finální úkol pak spočíval v implementaci exportu naměřených dat do vhodného souboru. Jako nejvhodnější formát byl zvolen CSV, který má jednoduchou strukturu sloužící k výměně tabulkových dat. Ta můžeme dále zpracovat např. některým z tabulkových procesorů, jež většinou zmíněný formát podporují.

Komunikace s databází

Pro implementaci částí modulů aplikace komunikujících s databází byla použita technologie tzv. odpojené vrstvy ADO.NET. Odpojený scénář je založen na principu připojení k DB jen na nezbytně nutnou dobu potřebnou pro načtení či aktualizaci dat. V tomto omezeném čase aplikace ze získaných dat z databáze vytvoří obraz datového zdroje a uloží jej do paměti zařízení. K odpojeným datům pak lze přistupovat objektově. S ohledem na ostatní součásti aplikace, jež jsou též napsány v objektovém jazyce, je možnost objektového přístupu k DB

velkou výhodou. Pro odpojenou vrstvu jsou stěžejní 3 třídy – *DataSet, DataTable* a *DataAdapter.*

Třída *DataSet* reprezentuje odpojený datový zdroj. Do tohoto zdroje je možné přidat databázové tabulky, uložené funkce a procedury databáze, ale i jednotlivé relace mezi tabulkami, nebo objekty nepocházející z databáze – např. kolekce dat definovaných třídami nebo CSV soubory. *DataSet* tyto zdroje převede na kolekci databázových tabulek *DataTable.*

Struktura třídy *DataTable* se podobá fyzické databázové tabulce. Obsahuje kolekce sloupců, reprezentujících strukturu dat tabulky, i řádků, tedy jednotlivých záznamů v těchto tabulkách. Třída dále implementuje metody pro práci s daty obou kolekcí.

Poslední zmíněnou třídou je *TableAdapter*. Ten implementuje metody pro načítání dat z databáze do odpojeného scénáře (např. třídy *DataSet*), nebo pro aktualizaci obrazu dat upraveného odpojenou aplikací ve zdrojové DB.

Modul Závody využívá datového zdroje implementovaného jako samostatný projekt, jenž je sdílený též s modulem Správa databáze. Samotný odpojený scénář je vytvořen pomocí Návrháře datových sad, který je součástí sady Microsoft Visual Studio. Při přidávání objektů do scénáře byl využit průvodce, který nám pomůže s výběrem přidávaných objektů (např. tabulek nebo uložených procedur či zadaných SQL příkazů z databáze) a automaticky k nim vytvoří příslušné instance tříd *DataTable* a *TableAdapter,* jež reprezentují data zdrojového objektu. Nechceme-li vracet údaje v datové tabulce, je možné automaticky z vybrané procedury či SQL dotazu návrhářem vygenerovat metodu se vstupními i výstupními parametry. Tuto metodu pak můžeme zavolat přes instanci příslušného *TableAdapteru*. Zmíněné možnosti bylo využito především u dotazů, kterým vyhovuje vždy pouze 1 záznam z databázové tabulky. Takový lze jednoduše vrátit prostřednictvím výstupních parametrů vzniklé metody.

Všechny datové sady používané v aplikaci byly vytvořeny na základě příslušných uložených procedur. Vzniklá třída *DataSet* obsahuje 33 datových zdrojů vracejících data příslušné databázové tabulky. Vygenerované *TableAdapter* třídy příslušných datových zdrojů obsahují implicitně metodu *Fill*, sloužící k načtení dat z fyzického zdroje. K některým tabulkám byly ještě návrhářem, opět na základě uložených procedur, dogenerovány další metody, které slouží pro úpravu dat a získávání vybraných hodnot fyzické databáze. Grafický návrh implementovaného datového zdroje je uveden v příloze této bakalářské práce.

Obsluha modulu

Po spuštění modulu máme v nabídce v horní části formuláře na výběr ze tří karet – *Měření, Listiny* a *Závody.* Jsou-li k dispozici taktéž potřebná data v příslušných tabulkách databáze

a máme-li k dispozici i časy zaznamenané modulem Časomíra, můžeme přistoupit k vyhodnocení výsledků daného závodu.

V první řadě se přesuneme na kartu *Závody*, kde je nutné vyplnit informace o měřeném klání. To můžeme provést více způsoby. Můžeme přidat nový závod, kde vyplníme všechny požadované údaje. Pokud však již databáze nějaké závody obsahuje, můžeme z nich tyto informace zkopírovat, a to včetně startovních pořadí závodníků, kteří se klání, z něhož data kopírujeme, účastnili. Po tomto kroku se přesuneme na kartu *Listiny*. Pokud jsme však údaje včetně startovních pořadí zkopírovali z jiného závodu a nemáme v úmyslu startovní listinu dále upravovat, můžeme tento krok přeskočit.

Na kartě *Listiny* nejprve vybereme příslušný závod, u něhož chceme měnit startovní pořadí. K dispozici máme i filtr pro zobrazení jen některých soutěžících. V dolní polovině formuláře máme na výběr ze dvou karet – *Startovní listina* a *Výsledková listina*. Na první zmíněné kartě máme k dispozici ovládací prvky pro editaci soutěžících v závodě – Můžeme je přidávat, upravovat, odebírat, měnit jejich startovní pořadí. Po přidání všech soutěžících můžeme vygenerovat startovní listinu. Na panelu nástrojů máme možnost zobrazit nebo skrýt některé sloupce. Ty můžeme taktéž mezi sebou různě přesouvat „uchopením a přesunem“ za záhlaví konkrétního sloupečku. Po dokončení potřebných změn klikneme na tlačítko „Exportovat listinu.“ Budeme dotázáni na umístění výsledného souboru. Poté se již vybraná data exportují v té samé podobě, jaká byla k vidění v příslušném ovládacím prvku, do CSV souboru.

Následně přejdeme na kartu *Měření*. Zde opět vybereme odpovídající závod a stiskneme tlačítko „Měřit.“ Poté by se nám již měla v příslušné tabulce zobrazit startovní pořadí, která můžeme ještě poupravit dle nabízeného filtru. Startovní čísla však zůstanou zachována. Pro správnou funkčnost musíme mít zobrazený též modul Časomíra, ve kterém musí být k dispozici naměřená data. Pokud je tento krok splněn, zobrazí se nám příslušné pokusy v odpovídající tabulce karty *Měření.* Zde nemusí být viditelné všechny naměřené výkony. Ty se totiž filtrují podle odpovídajícího typu závodu uvedeného v DB a typu naměřeného pokusu (u závodu PÚ budou zobrazeny pouze časy požárních útoků atd.). Na panelu nástrojů je též k dispozici rozšířená možnost filtrování dat. Můžeme např. skrýt pokusy označené jako nevyhodnocované nebo zobrazit ty, které již byly v daném závodě některému soutěžícímu přiřazeny.

Znázorněme si poslední možnost v následující ukázce kódu. Pomocí metody *Fill* příslušného *TableAdapteru* jsou nejprve načtena data z DB. Poté jsou všechny řádky tabulky *DataTable* procházeny cyklem. V jedné iteraci jsou vždy nejprve zjištěny potřebné údaje –číslo měření a časy startu a konce pokusu. Tyto informace jsou následně pomocí LINQ hledány

```
//Načte z DB naměřené pokusy
namerenePokusyTableAdapter.Fill(casomiraDBDataSet.NamerenePokusy, zavodID);
//Pokud si uživatel nepřeje zobrazit již naměřené pokusy z časomíry
//a zároveň již byly nějaké pokusy do DB přiřazeny
if (!prirazenePokusyMenuItem.Checked && casomiraDBDataSet.NamerenePokusy.Count > 0)
{
    int? cisloMereni;
    DateTime? startPokusu, konecPokusu;

    List<Pokus> prvkyKeSmazani = new List<Pokus>();
    foreach (var radek in casomiraDBDataSet.NamerenePokusy.AsEnumerable())
    {
        cisloMereni = radek[1] as int?;
        startPokusu = radek[2] as DateTime?;
        konecPokusu = radek[3] as DateTime?;

        //řádek tabulky byl v DB změněn a hodnota dat. typu neodpovídá očekávané
        if (!(cisloMereni is int?) || !(startPokusu is DateTime?)
            || !(konecPokusu is DateTime?)) break;

        //pokusí se najít pokusy se zadaným číslem měření, startem a koncem pokusu jako již
        //v DB existující a odstraní je z výběru.
        IEnumerable<Pokus> nalezeneDuplicity =
            from pokus in dataCasomira
            where pokus.CisloMereni == cisloMereni && pokus.StartPokusu == startPokusu
            && pokus.KonecPokusu == konecPokusu
            select pokus;
        prvkyKeSmazani.AddRange(nalezeneDuplicity);
    }
    //vyfiltruje nevyhovuující pokusy
    dataCasomira = dataCasomira.Except(prvkyKeSmazani.AsEnumerable());
}
```

Obrázek 23 – Ukázka C# kódu – Filtrování kolekce pokusů s využitím LINQ

v kolekci naměřených pokusů z časomíry. Je-li nalezena shoda, jsou tyto pokusy přidány do seznamu prvků k vyloučení. Po skončení cyklu jsou v tomto seznamu již všechny nalezené pokusy ke smazání, které jsou následně z kolekce všech pokusů z časomíry vyloučeny.

Poté vybereme pokus v obou tabulkách a výběrem příslušné možnosti přiřadíme. Selekce obou tabulek jsou spárovány. V případě výběru rozdílných čísel řádků tak musíme upravit posun výběru. Po přiřazení všech pokusů příslušným tlačítkem ukončíme měření a můžeme přistoupit k vytvoření výsledkových listin.

Přesuneme se zpět na kartu *Listiny*. Zde vybereme nabízenou kartu *Výsledkové listiny*. Opět upravíme filtr soutěžících podle potřeby a též si zobrazíme, případně uspořádáme nabízené sloupce tabulky. Výsledková listina by se nám po zmíněných krocích a obsluhou tlačítka „Zobrazit listiny“ měla automaticky vygenerovat. Poté již můžeme výslednou tabulku exportovat do CSV souboru.

5.4.3 Modul Správa databáze

Jak jsme se mohli dozvědět z kapitoly 5.2, modul Správa databáze je určen pro modifikaci dat tabulek v databázi. Jedná se o vkládání, mazání a úpravu zmíněných dat. Modul neslouží k úpravě všech tabulek, ale pouze těch, jež přímo nesouvisí se závody. Správu takových tabulek má totiž na starost součást aplikace, o které byla řeč v předchozí podkapitole (5.4.2).

Komunikace s databází

Pro komunikaci s databází je opět využita byla použita technologie tzv. odpojené vrstvy ADO.NET. Více informací o odpojeném scénáři je uvedeno v kapitole 5.4.2, část Komunikace s databází.

Správa databázových tabulek

Po spuštění modulu se nám v horní části formuláře zobrazí několik karet. Název každé karty pak reprezentuje příslušné jméno skutečné databázové tabulky. Výběrem příslušné karty docílíme zobrazení záznamů odpovídající tabulky prostřednictvím GUI. Jak již bylo řečeno, data můžeme přidávat, upravovat a odstraňovat. Docílíme toho výběrem příslušného záznamu (u odstranění můžeme vybrat i více řádků najednou) a obsluhou tlačítka na panelu nástrojů či výběrem volby v místní nabídce, jež se zobrazí po kliknutí na tabulku pravým tlačítkem myši. Předpokládejme, že chceme záznam v databázi upravit.

Po provedení potřebných kroků se uživateli zobrazí příslušné dialogové okno. Do jeho ovládacích prvků jsou před zobrazením načtena existující data z tabulky. Pojďme si tuto část kódu programu, ošetřujícího zmíněnou činnost, představit.

```
//má-li ID hodnotu (pouze u editace dat, u vkládání se přeskakuje)
if (ID.HasValue)
{
    //Načte data z DB dle ID získaného z příslušného řádku tabulky
    tymyTableAdapter.DejTym(ID, out string nazevTymu, out int? soutezniKategorieID, out int? okresID, out int? hasicskySborID,
            out string jmenoVedouci, out string prijmeniVedouci, out long? telefonVedouci, out string emailVedouci);

    //Veřejné vlastnosti formuláře
    HasicskySborID = hasicskySborID;
    OkresID = okresID;

    //Doplní údaje do příslušných ovládacích prvků
    nazevTymuTextBox.Text = nazevTymu;
    VyberHodnotuIDComboBox(soutezniKategorieComboBox, soutezniKategorieID);

    jmenoVedouciTextBox.Text = jmenoVedouci;
    prijmeniVedouciTextBox.Text = prijmeniVedouci;
    telefonVedouciTextBox.Text = TelefonniCisloToString(telefonVedouci);
    emailVedouciTextBox.Text = emailVedouci;
}
```

Obrázek 24 – Ukázka C# kódu – Naplnění ovládacích prvků údaji z databáze

Uvedený kód je součástí ošetřující metody události *OnLoad*, jež se vyvolá před prvním zobrazením formuláře příslušného dialogového okna. Metoda *DejTym*, zobrazená v ukázce, je

generovaná automaticky z uložené procedury databáze přes návrhář datové sady odpojené vrstvy ADO.NET. Dotaz SQL uložené procedury pak vypadá následovně:

```
USE [CasomiraDB]
GO
/****** Object:  StoredProcedure [dbo].[DejTym]    Script Date: 24.05.2020 18:41:33 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[DejTym]  @ID int = NULL, @Nazev nvarchar(MAX) = NULL OUTPUT, @SouteznikategorieID int = NULL OUTPUT, @OkresID int = NULL OUTPUT,
                                @HasicskySborID int = NULL OUTPUT, @JmenoVedouci nvarchar(MAX) = NULL OUTPUT, @PrijmeniVedouci nvarchar(MAX) = NULL OUTPUT,
                                @TelefonVedouci bigint = NULL OUTPUT, @EmailVedouci nvarchar(MAX) = NULL OUTPUT
AS
--Výběrový dotaz, který odpovídající údaje přiřadí do parametrů definovaných výše
SELECT  @Nazev = Nazev, @SouteznikategorieID = Soutezni_kategorie_ID, @OkresID = Okres_ID, @HasicskySborID = Hasicsky_sbor_ID,
        @JmenoVedouci = Jmeno_vedouci, @PrijmeniVedouci = Prijmeni_vedouci, @TelefonVedouci = Telefon_vedouci, @EmailVedouci = Email_vedouci
FROM    Tymy
WHERE  ID = @ID
```

Obrázek 25 – Ukázka SQL kódu – Uložená procedura pro výběr záznamu a předání údajů výstupními parametry

Po zobrazení dialogového okna provedeme úpravy dat v ovládacích prvcích a klikneme na tlačítko „OK.“ Záznam je následně uložen do databáze a dialogové okno je uzavřeno.

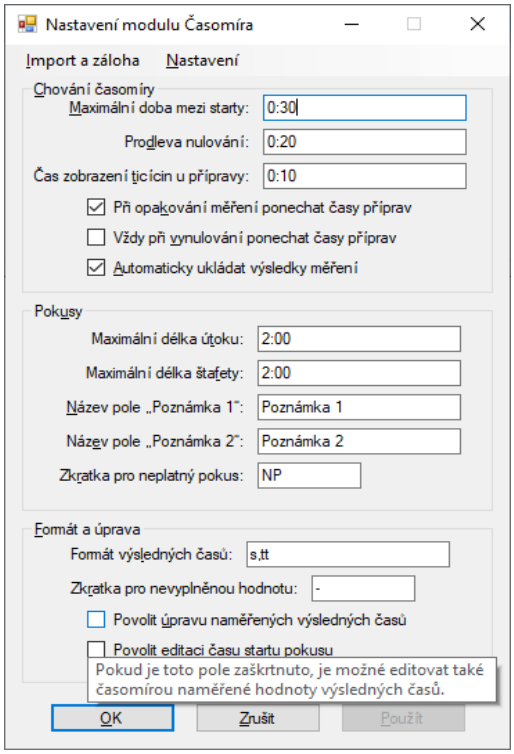
Někdy však může nastat problém, např. při vkládání duplicitních dat do DB, jež není povoleno, nebo při zadání údaje v nepovoleném formátu. V takovém případě se uživateli zobrazí další dialogové okno s textem chyby. Od obsluhy je poté očekáváno potvrzení zprávy o nedokonalosti a nutná korekce dat v dialogovém okně pro úpravu záznamu tabulky.

5.5 Nastavení aplikace

Aplikace umožňuje měnit určité parametry, které ovlivňují chování programu. Hlavní nastavení programu je zakomponováno v modulu Časomíra. Lze v něm upravit nejen chování třídy *Casomira*, zajišťující zpracování výsledků z fyzické časomíry, ale i např. formát časů výsledných pokusů nebo další možnosti. Změna nastavení probíhá přes dialogové okno, které umožňuje zálohovat a obnovit hodnoty nastavení do/ze souboru. Aplikace si pamatuje hodnoty nastavení i po jejím ukončení. Při opětovném spuštění budou načteny poslední známé možnosti programu.

Je-li otevřen modul Časomíra, nastavení platí taktéž pro modul Závody, který z těchto možností přebírá především podobu výsledných časů pokusů. V opačném případě je pro druhý zmíněný modul použito výchozího nastavení.

Podoba dialogového okna nastavení aplikace je uvedena v následující ukázce. Příslušný formulář Windows Forms v sobě též obsahuje „mininávod," který se zobrazí při najetí ukazatelem myši na příslušný popisek ovládacího prvku sloužícího ke změně dané možnosti.



Obrázek 26 – Formulář nastavení modulu Časomíra

Také samotný modul Závody nějaká nastavení obsahuje. Ta nalezneme v nabídce stavového řádku příslušné miniaplikace. Kromě schopností skrývání některých sloupců tabulky se jedná o možnost automatického přidělování poznámek k naměřeným pokusům z časomíry. Tato změna se samozřejmě promítne i do stejnojmenného modulu. Pokud je taková možnost povolena, při přiřazení pokusu jsou spolu s přesným datem přidělení do poznámek připsány údaje o soutěžícím, jemuž daný pokus „patří". Tyto informace se mohou hodit např. při zpětném zjišťování chybného přidělení naměřeného výkonu jinému soutěžícímu apod.

Moduly Časomíra a Závody si také ukládají nastavení rozložení a šířek sloupců ve všech použitých ovládacích *DataGridView* a *ListView*. Tyto hodnoty jsou při opětovném spuštění zmíněných miniaplikací načteny z dřívějška (existují-li).

5.6 Instalace a konfigurace aplikace a potřebných součástí

Před prvním použitím programu je pro zajištění jeho bezchybné funkčnosti nutné nainstalovat potřebný software a konfigurovat aplikaci.

5.6.1 Instalace aplikace a potřebných součástí

Firesport Timekeeper

Aplikace Firesport Timekeeper nevyžaduje zvláštní instalaci. K jejímu používání stačí přesunout soubory z příslušné složky, která je součástí přiloženého CD, do počítače a spustit program *Firesport_Timekeeper.exe*. Chceme-li též využívat moduly spolupracující s databází, je nutné aplikaci před prvním použitím nakonfigurovat.

Microsoft SQL Server

Nemáme-li možnost využití databázového serveru a chceme-li využívat aplikaci také k vyhodnocování výsledků, je nutné do příslušného počítače nainstalovat Microsoft SQL Server. Nejlepší volbou je sáhnout po aktuální verzi MSSQL Express 2019, která je k dispozici bezplatně např. na webu Microsoft. Dále se řídíme pokyny instalačního programu. Po nainstalování je nutné SQL Server spustit a importovat do něho databázi z přiloženého CD.

Microsoft SQL Server Management Studio 18

Instalujeme-li do počítače též Microsoft SQL Server, pro správu databáze může být užitečný program Microsoft SQL Server Management Studio, který je taktéž poskytován zadarmo. Poslední verzi této aplikace je možné stáhnout opět ze stránek Microsoftu.

5.6.2 Konfigurace aplikace a potřebných součástí

Firesport Timekeeper

Chceme-li software používat k vyhodnocení výsledků, je nutné před prvním použitím nakonfigurovat připojení k databázi. To provedeme úpravou příslušných údajů v souboru *Databaze.dll.config,* který nalezneme ve složce s aplikací. Zde upravíme vlastnost *connectionString*. Dáváme pozor na zachování struktury souboru, změnu dat provádíme pouze v uvozovkách u příslušné vlastnosti. Více informací o syntaxi připojovacího řeztězce ConnectionString nalezneme v online dokumentaci .NET.

Microsoft SQL Server

Aplikace kromě spuštění služby *SQL Server (SQLEXPRESS)* nevyžaduje zvláštní konfiguraci. Spuštění služby provedeme přes komponentu SQL Server Configuration Manager.

Databáze

Abychom mohli pracovat s databází, je nutné ji nejdříve importovat na Microsoft SQL Server. K tomu nám nejlépe poslouží program Microsoft SQL Server Management Studio. Otevřeme jej a nastavíme připojení k příslušnému SQL Serveru. Po tomto kroku klikneme pravým tlačítkem myši na podsložku *Databases* příslušného SQL serveru. Tyto složky nalezneme v průzkumníku objektů *Object Explorer*. Poté vybereme volbu *Import data Tier-Application.* Průvodce nás provede importem databáze ze souboru s příponou *.bacpac,* který nalezneme v přiloženém CD. Na konci průvodce databázi pojmenujeme a dokončíme import. Název databáze je též nutné změnit v ConnectionStringu aplikace k vyhodnocení výsledků (viz Konfigurace aplikace Firesport Timekeeper).

ZÁVĚR

Hlavním cílem bakalářské práce bylo vytvořit software určený pro OS Windows, který bude zpracovávat naměřené výsledky z časomíry pro požární sport. Program by při měření neměl vyžadovat žádnou, nebo jen minimální obsluhu. Aplikace měla být vyvíjena v programovacím jazyce C#.

Nejprve bylo nutné nastudovat si sériovou komunikaci mezi fyzickou časomírou a počítačem. Zde nastal menší problém. Reálná časomíra totiž nedokáže rozpoznat, tudíž ani nemůže PC informovat o nastalém neplatném pokuse, který se projeví vynulováním časomíry před ukončením měření všech pokusů cílovými prvky daných drah. Vše bylo nakonec vyřešeno v aplikaci pomocí výpočtu rozhodující doby od startu po vynulování časomíry. Algoritmus následně spočítanou hodnotu porovná s maximální přípustnou dobou a rozhodne, zda je pokus neplatný, nebo se jedná o opakovaný start. Následoval vývoj stěžejní miniaplikace Časomíra.

Po jejím zdárném dokončení bylo nutné program řádně otestovat. Stalo se tak na soutěži TFA (Železný hasič) konané 7. 12. 2019 v sokolovně v Horní Čermné. Aplikace uspěla bez chyby a pomohla k urychlení vyhodnocení pokusů bezmála 60 závodníků napříč 6 soutěžními kategoriemi. Tehdy však ještě SW postrádal modul pro vyhodnocení výsledků. Zpracování naměřených časů tak bylo v tomto případě realizováno v tabulkovém procesoru.

Po úspěšném otestování hlavního modulu následoval návrh databáze pro vyhodnocení výsledků, který se obešel bez komplikací. Problémy nastaly až s tvorbou aplikací komunikujících s touto databází. To pro mne byla v podstatě nová věc. Implementace modulů Závody a Výsledkové listiny tak byla časově velmi náročná, neboť jsem se při ní stále učil novým a novým věcem. Následné testování obou modulů, které tentokrát kvůli koronavirové krizi proběhlo pouze v domácích podmínkách, však dopadlo vesměs dobře. Bylo sice odhaleno pár menších chyb v programu, nejednalo se ale naštěstí o nic závažného. Nedostatky tak byly brzy opraveny. Při další domácí zkoušce již výsledná aplikace uspěla na výbornou.

Všechny cíle bakalářské práce byly splněny. Teoretická část práce se zabývá disciplínami požárního sportu a následně řeší samotné měření těchto disciplín. V další kapitole dílo popisuje princip komunikace PC se zařízeními. Poté jsou již popsány použité technologie a samozřejmě vlastní návrh a implementace aplikace, která je praktickou částí bakalářské práce. Vyvinutý program zpracovává časy z časomíry a umožňuje je přiřadit soutěžícím s konkrétními startovními pořadími v jednotlivých závodech. Program při měření vyžaduje pouze minimální obsluhu.

Původně bylo myšleno, že by i přiřazování naměřených pokusů soutěžícím v databázi bylo plně automatické. Tento krok nakonec nebyl realizován. Nebyl by problém v technické obtížnosti, naprogramovat přiřazení právě naměřeného pokusu do databáze podle zadané startovní listiny je vcelku jednoduché. Sporná otázka však nastává při identifikaci závodníka. Ne vždy totiž může měření probíhat podle startovní listiny, ba naopak se stává, že si někteří závodníci po schválení rozhodčím dodatečně vymění svá pořadí na startu. Takovou situaci však časomíra ani program nerozpozná. Při automatickém přiřazení bez úpravy startovního pořadí by tedy došlo k chybě – pokus by nebyl přidělen jeho „pravému majiteli." To je nevhodné, neboť obsluha aplikace při troše nepozornosti nemusí na toto pochybení přijít. Přiřazení pokusů soutěžícím v DB je tak plně v režii obsluhy programu. Samotný výběr nejpravděpodobnějších dvojic pokus-soutěžící naštěstí probíhá automaticky, takže pověřená osoba jej v podstatě pouze potvrdí, případně provede korekci. I přesto však byla zadaná kritéria pro vypracování bakalářské práce dodržena.

Vývoj aplikace touto bakalářskou prací nekončí. Software čeká vylepšení po jeho řádném odzkoušení v reálném provozu a následném nabytí dostatečného množství cenných poznatků. Do budoucna je plánováno vytvořit webovou aplikaci, která by umožnila online zobrazování průběžných výsledků právě měřených soutěží a samozřejmě i historických statistik všech uplynulých klání.

Bakalářská práce pro mne byla přínosem. Při její tvorbě jsem si podrobněji nastudoval pravidla požárního sportu, zopakoval si základy objektově orientovaného programování a také tvorby relačních databází i SQL dotazů. Díky této práci jsem se bezesporu naučil mnoha novým věcem především z tvorby databázových aplikací. Nabyté zkušenosti v budoucnu určitě využiji.

POUŽITÁ LITERATURA

[1] Disciplíny požárního sportu. *Hasičský záchranný sbor České republiky* [online]. Praha: Generální ředitelství Hasičského záchranného sboru ČR, 2019 [cit. 2020-04-23]. Dostupné z: https://www.hzscr.cz/clanek/discipliny-pozarniho-sportu.aspx

[2] Historie požárního sportu. *Hasičský záchranný sbor České republiky* [online]. Praha: Generální ředitelství Hasičského záchranného sboru ČR, 2019 [cit. 2020-04-26]. Dostupné z: https://www.hzscr.cz/clanek/historie-pozarniho-sportu.aspx

[3] Disciplíny požárního sportu: Běh na 100 m s překážkami. *Hasičský záchranný sbor České republiky* [online]. Praha: Generální ředitelství Hasičského záchranného sboru ČR, 2019 [cit. 2020-04-26]. Dostupné z: https://www.hzscr.cz/clanek/discipliny-pozarniho-sportu.aspx?q=Y2hudW09Mg%3d%3d

[4] *Sbírka interních aktů řízení generálního ředitele HZS ČR - částka 10/2018: Příloha k Pokynu GŘ HZS ČR č. 10/2018 - Pravidla požárního sportu*. Praha, 2018. Dostupné z: https://www.hzscr.cz/soubor/pravidla-ps-2018-pdf.aspx

[5] Disciplíny požárního sportu: Výstup do 4. podlaží cvičné věže. *Hasičský záchranný sbor České republiky* [online]. Praha: Generální ředitelství Hasičského záchranného sboru ČR, 2019 [cit. 2020-04-26]. Dostupné z: https://www.hzscr.cz/clanek/discipliny-pozarniho-sportu.aspx?q=Y2hudW09Mw%3d%3d

[6] Disciplíny požárního sportu: Štafeta 4 x 100 m s překážkami. *Hasičský záchranný sbor České republiky* [online]. Praha: Generální ředitelství Hasičského záchranného sboru ČR, 2019 [cit. 2020-04-27]. Dostupné z: https://www.hzscr.cz/clanek/discipliny-pozarniho-sportu.aspx?q=Y2hudW09NA%3d%3d

[7] Disciplíny požárního sportu: Požární útok. *Hasičský záchranný sbor České republiky* [online]. Praha: Generální ředitelství Hasičského záchranného sboru ČR, 2019 [cit. 2020-04-27]. Dostupné z: https://www.hzscr.cz/clanek/discipliny-pozarniho-sportu.aspx?q=Y2hudW09NQ%3d%3d

[8] RS-232. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2020-05-12]. Dostupné z: https://cs.wikipedia.org/wiki/RS-232

[9] TIŠNOVSKÝ, Pavel. Co se děje v počítači: Sériový port RS-232C. *Root.cz* [online]. 2008, **2008**(11), 1-7 [cit. 2020-05-12]. ISSN 1212-8309. Dostupné z: https://www.root.cz/clanky/seriovy-port-rs-232c/

[10] Universal Serial Bus. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2020-05-12]. Dostupné z: https://cs.wikipedia.org/wiki/Universal_Serial_Bus

[11] PAVLIS, Jakub. USB 3.1 a rozdíl mezi Gen1 a Gen2. *Notebook.cz* [online]. 2017, **2017**(4), 1 [cit. 2020-05-12]. ISSN 1214-2875. Dostupné z: https://notebook.cz/clanky/technologie/2017/usb-3-1

[12] TIŠNOVSKÝ, Pavel. Co se děje v počítači: Universální sériová sběrnice (USB). *Root.cz* [online]. 2009, **2009**(1), 1-9 [cit. 2020-05-12]. ISSN 1212-8309. Dostupné z: https://www.root.cz/clanky/universalni-seriova-sbernice-usb/

[13] Ethernet. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2020-05-13]. Dostupné z: https://cs.wikipedia.org/wiki/Ethernet

[14] BURIAN, Pavel. *Internet inteligentních aktivit*. 1. vyd. Praha: Grada, 2014. ISBN 978-802-4790-763.

[15] Wi-Fi. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2020-05-13]. Dostupné z: https://cs.wikipedia.org/wiki/Wi-Fi

[16] .NET. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2020-05-15]. Dostupné z: https://cs.wikipedia.org/wiki/.NET

[17] Úvod do .NET Frameworku. *DotNETPortal.cz* [online]. Liberec: Tomáš Jecha, 2009 [cit. 2020-05-15]. Dostupné z: https://www.dotnetportal.cz/clanek/125/Uvod-do-NET-Frameworku

[18] C Sharp. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2020-05-15]. Dostupné z: https://cs.wikipedia.org/wiki/C_Sharp

[19] ČÁPKA, David. Základní konstrukce jazyka C#: Lekce 1 - Úvod do C# a .NET frameworku. *ITnetwork.cz* [online]. 2012, **2012**(1), 1 [cit. 2020-05-16]. ISSN 2464-6326. Dostupné z: https://www.itnetwork.cz/csharp/zaklady/c-sharp-tutorial-uvod-do-jazyka-a-dot-net-framework

[20] AUGUSTÝN, Michal. LINQ a lambda expressions. *Zdroják.cz* [online]. 2009, **2009**(10), 1 [cit. 2020-05-15]. ISSN 1803-5620. Dostupné z: https://www.zdrojak.cz/clanky/linq-a-lambda-expressions/

[21] LACKO, Ľuboslav. *1001 tipů a triků pro SQL*. 1. vyd. Brno: Computer Press, 2011. ISBN 978-80-251-3010-0.

[22] LACKO, Ľuboslav. *Mistrovství v SQL Server 2012: Kompletní průvodce databázového experta*. 1. vyd. Brno: Computer Press, 2013. ISBN 978-80-251-3773-4.

[23] Editions and supported features of SQL Server 2019 (15.x). *Microsoft: Docs* [online]. Redmond (WA), USA: Microsoft Corporation, 2020 [cit. 2020-05-16]. Dostupné z: https://docs.microsoft.com/en-us/sql/sql-server/editions-and-components-of-sql-server-version-15?view=sql-server-ver15

[24] Download SQL Server Management Studio (SSMS). *Microsoft: Docs* [online]. Redmond (WA), USA: Microsoft Corporation, 2020 [cit. 2020-05-16]. Dostupné z: https://docs.microsoft.com/cs-cz/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15

[25] Microsoft Visual Studio. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2020-05-16]. Dostupné z: https://cs.wikipedia.org/wiki/Microsoft_Visual_Studio

[26] ŽŮREK, Michal. Visual Studio: Lekce 1 - Visual Studio - Úvod do vývojového prostředí. *ITnetwork.cz* [online]. 2015, **2015**(1), 1 [cit. 2020-05-16]. ISSN 2464-6326. Dostupné z: https://www.itnetwork.cz/csharp/visual-studio/tutorial-visual-studio-uvod

[27] Porovnání edic sady Visual Studio 2019. *Microsoft: Visual Studio* [online]. Redmond (WA), USA: Microsoft Corporation, 2020 [cit. 2020-05-16]. Dostupné z: https://visualstudio.microsoft.com/cs/vs/compare/

[28] LCD časomíra LV 4,66 - 4 vstupy. *Časomíry Ing. Libor Valeš* [online]. Žákava: Ing. Libor Valeš, 2020 [cit. 2020-05-18]. Dostupné z: http://www.casomiry.com/index.php?casomiry/LCD466_4

[29] *Interní informace od výrobce časomír LV, Ing. Libora Valeše*. Žákava, 2019.

[30] Obrázek - Běh na 100 m s překážkami. In: *Mistrovství ČR v požárním sportu 2019* [online]. Ústí nad Labem: Hasičský záchranný sbor Ústeckého kraje, 2019 [cit. 2020-04-26]. Dostupné z: https://www.mrps2019.cz/wp-content/uploads/photo-gallery/imported_from_media_libray/P-MRPS75.jpg?bwg=1567232712

[31] Obrázek - Výstup do 4. podlaží cvičné věže. In: *Mistrovství ČR profesionálních a dobrovolných hasičů v požárním sportu 2017 – Praha 24. – 27. srpna 2017* [online]. Praha: Hasičský záchranný sbor hl. m. Prahy, 2017 [cit. 2020-04-26]. Dostupné z: http://mrps2017.hasicipraha.cz/wp-content/gallery/vez-muzi/20170825_16-03-50_0017.JPG

[32] Obrázek - Domeček - Štafeta 4 x 100 m s překážkami. In: *Mistrovství ČR profesionálních a dobrovolných hasičů v požárním sportu 2017 – Praha 24. – 27. srpna 2017* [online]. Praha: Hasičský záchranný sbor hl. m. Prahy, 2019 [cit. 2020-04-27]. Dostupné z: http://mrps2017.hasicipraha.cz/wp-content/gallery/stafeta-sdh-muzi/20170825_16-29-35_0499.JPG

[33] Obrázek - Hašení zapálené hořlavé směsi - Štafeta 4 x 100 m s překážkami. In: *Mistrovství ČR profesionálních a dobrovolných hasičů v požárním sportu 2017 – Praha 24. – 27. srpna 2017* [online]. Praha: Hasičský záchranný sbor hl. m. Prahy, 2017 [cit. 2020-04-27]. Dostupné z: http://mrps2017.hasicipraha.cz/wp-content/gallery/stafeta-hzs/20170825_17-51-13_0521.JPG

[34] Obrázek - Sestavování dopravního vedení - Požární útok. In: *Fotky Google* [online]. Praha: Hasičský záchranný sbor hl. m. Prahy, 2017 [cit. 2020-04-27]. Dostupné z: https://photos.google.com/share/AF1QipMGEmmhkvjy47orudg4FQsKbg7l7T8R8mIDi3205iPrguWFnBYfjOqpJ_m3h4RUEw/photo/AF1QipMDigyAkrN_ZHKZ_c9glfegRcQeEztgcZ-TDUEJ?key=TmxkVHp3R25zOW5ncWF3SUhmd0o1SnFULXBIb1FB

[35] Obrázek - Nástřik terčů - Požární útok. In: *Mistrovství ČR profesionálních a dobrovolných hasičů v požárním sportu 2017 – Praha 24. – 27. srpna 2017* [online]. Praha: Hasičský záchranný sbor hl. m. Prahy, 2017 [cit. 2020-04-27]. Dostupné z: http://mrps2017.hasicipraha.cz/wp-content/gallery/pozarni-utok/20170827_08-42-44_0013.JPG

[36] Obrázek - Startovací zařízení - Optická závora. In: *MIDUR STORE - Komplexní sportovní vybavení* [online]. Klátová Nová Ves, SR: Midur, s.r.o., 2019 [cit. 2020-05-06]. Dostupné z: https://www.midur.eu/fotky41707/fotos/_vyr_8567322_z1-1.jpg

[37] Obrázek - Startovací zařízení - Startovací pistole. In: *TRV Elektronik* [online]. Lesní Hluboké: Tomáš Kocáb - TRV elektronik, 2019 [cit. 2020-05-06]. Dostupné z: https://www.trv-kocab.cz/images/stories/virtuemart/product/startovaci-pistol-9mm.jpg

[38] Obrázek - Nástřikový terč. In: *Požární bezpečnost s.r.o.: VYZBROJNA.cz* [online]. Jihlava: Požární bezpečnost s.r.o., 2007 [cit. 2020-05-06]. Dostupné z: https://www.vyzbrojna.cz/upload/foto/nastrikovy-terc-pro-pozarni-utok-15808-2.jpg

[39] *Nákresy a fotky terčů pro pořadatele závodů Extraligy ČR v požárním útoku*. Bruntál, 2019. Dostupné z: www.extraliga-pu.cz/phocadownloadpap/dokumenty/terce_new.pdf

[40] Obrázek - Nášlapná deska pro výstup na věž. In: *TRV Elektronik* [online]. Lesní Hluboké: Tomáš Kocáb - TRV elektronik, 2019 [cit. 2020-05-07]. Dostupné z: https://www.trv-kocab.cz/images/stories/virtuemart/product/naslapna-deska-pro-vystup-na-vez.jpg

[41] Obrázek - Konektor sériového portu D-Sub DE-9. In: *TIAO's Wiki* [online]. Los Angeles, Kalifornie, USA: NameCheap, Inc., 2008 [cit. 2020-05-08]. Dostupné z: https://www.tiaowiki.com/w/TIAO_USB_Multi_Protocol_Adapter_User%27s_Manual

[42] Obrázek - Sériová komunikace - Přenos 1 slova. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2020-05-08]. Dostupné z: https://sk.wikipedia.org/wiki/S%C3%A9riov%C3%BD_port#/media/S%C3%BAbor:Rs232_communication_(sk).svg

PŘÍLOHY

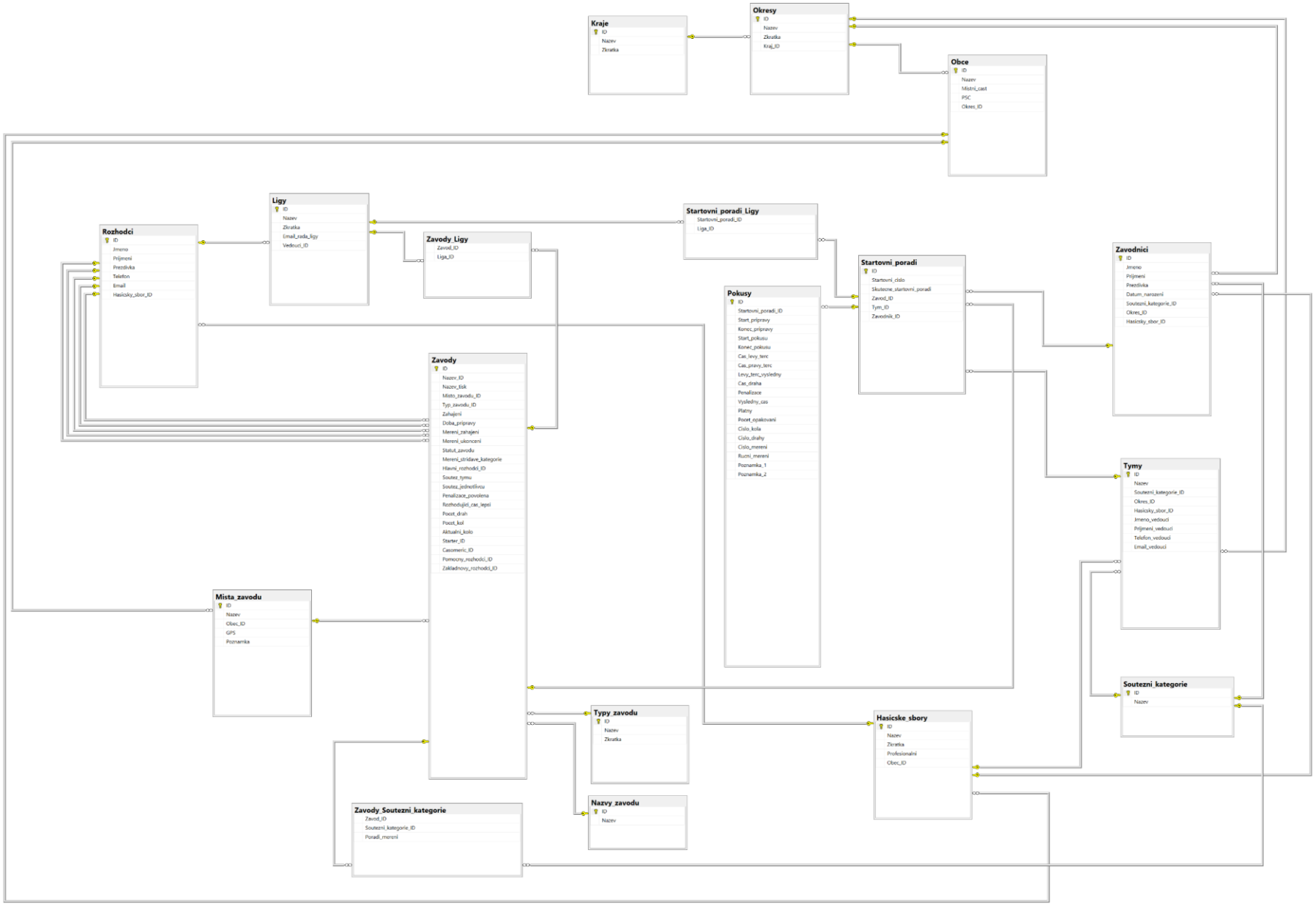
PŘÍLOHA A – ER DIAGRAM DATABÁZE „CASOMIRADB“

PŘÍLOHA B – NÁVRHY TABULEK DATABÁZE „CASOMIRADB“

PŘÍLOHA C – GRAFICKÝ NÁVRH DATOVÉHO ZDROJE ODPOJENÉ VRSTVY ADO.NET

PŘÍLOHA D – DIGITÁLNÍ PŘÍLOHA

PŘÍLOHA A – ER DIAGRAM DATABÁZE „CASOMIRADB“



PŘÍLOHA B – NÁVRHY TABULEK DATABÁZE „CASOMIRADB“

B.1 Tabulky *Zavody, Zavody_Ligy, Zavody_Soutezni_kategorie, Nazvy:zavodu, Typy_zavodu*

Column Name	Data Type	Allow Nulls
ID	int	☐
Nazev_ID	int	☐
Nazev_tisk	nvarchar(200)	☑
Misto_zavodu_ID	int	☑
Typ_zavodu_ID	int	☐
Zahajeni	datetime2(7)	☐
Doba_pripravy	float	☑
Mereni_zahajeni	datetime2(7)	☑
Mereni_ukonceni	datetime2(7)	☑
Statut_zavodu		☐
Mereni_stridave_kategorie	bit	☑
Hlavni_rozhodci_ID	int	☐
Soutez_tymu	bit	☐
Soutez_jednotlivcu	bit	☐
Penalizace_povolena	bit	☐
Rozhodujici_cas_lepsi	bit	☑
Pocet_drah	int	☐
Pocet_kol	int	☐
Aktualni_kolo	int	☑
Starter_ID	int	☐
Casomeric_ID	int	☑
Pomocny_rozhodci_ID	int	☑
Zakladnovy_rozhodci_ID	int	☑

Tabulka Zavody

Column Name	Data Type	Allow Nulls
Zavod_ID	int	☐
Soutezni_kategorie_ID	int	☐
Poradi_mereni	int	☑

Tabulka Zavody_Soutezni_kategorie

Column Name	Data Type	Allow Nulls
ID	int	☐
Nazev	nvarchar(200)	☐

Tabulka Nazvy_zavodu

Column Name	Data Type	Allow Nulls
Zavod_ID	int	☐
Liga_ID	int	☐

Tabulka Zavody_Ligy

Column Name	Data Type	Allow Nulls
ID	int	☐
Nazev	nvarchar(100)	☐
Zkratka	nvarchar(20)	☑

Tabulka Typy_zavodu

B.2 Tabulky *Mista_zavodu, Ligy, Hasicske_sbory, Rozhodci*

Column Name	Data Type	Allow Nulls
ID	int	☐
Nazev	nvarchar(200)	☐
Obec_ID	int	☐
GPS	geography	☑
Poznamka	nvarchar(MAX)	☑

Tabulka Mista_zavodu

Column Name	Data Type	Allow Nulls
ID	int	☐
Nazev	nvarchar(200)	☐
Zkratka	nvarchar(100)	☑
Profesionalni	bit	☐
Obec_ID	int	☐

Tabulka Hasicske_sbory

Column Name	Data Type	Allow Nulls
ID	int	☐
Nazev	nvarchar(200)	☐
Zkratka	nvarchar(10)	☐
Email_rada_ligy	nvarchar(150)	☑
Vedouci_ID	int	☑

Tabulka Ligy

Column Name	Data Type	Allow Nulls
ID	int	☐
Jmeno	nvarchar(100)	☐
Prijmeni	nvarchar(100)	☐
Prezdivka	nvarchar(100)	☑
Telefon	bigint	☐
Email	nvarchar(150)	☑
Hasicsky_sbor_ID	int	☐

Tabulka Rozhodci

B.3 Tabulky Pokusy, Startovni_poradi, Startovni_poradi_Ligy

Column Name	Data Type	Allow Nulls
ID	int	☐
Startovni_poradi_ID	int	☑
Start_pripravy	datetime2(7)	☑
Konec_pripravy	datetime2(7)	☑
Start_pokusu	datetime2(7)	☐
Konec_pokusu	datetime2(7)	☐
Cas_levy_terc	float	☑
Cas_pravy_terc	float	☑
Levy_terc_vysledny		☑
Cas_draha	float	☑
Penalizace	float	☑
Vysledny_cas		☑
Platny	bit	☐
Pocet_opakovani	int	☐
Cislo_kola	int	☐
Cislo_drahy	int	☑
Cislo_mereni	int	☐
Rucni_mereni	bit	☐
Poznamka_1	nvarchar(MAX)	☑
Poznamka_2	nvarchar(MAX)	☑

Tabulka Pokusy

Column Name	Data Type	Allow Nulls
ID	int	☐
Startovni_cislo	int	☐
Skutecne_startovni_poradi	int	☑
Zavod_ID	int	☐
Tym_ID	int	☑
Zavodnik_ID	int	☑

Tabulka Startovni_poradi

Column Name	Data Type	Allow Nulls
Startovni_poradi_ID	int	☐
Liga_ID	int	☐

Tabulka Startovni_poradi_Ligy

B.4 Tabulky *Tymy, Zavodnici, Soutezni_kategorie*

Column Name	Data Type	Allow Nulls
ID	int	☐
Nazev	nvarchar(200)	☐
Soutezni_kategorie_ID	int	☐
Okres_ID	int	☑
Hasicsky_sbor_ID	int	☑
Jmeno_vedouci	nvarchar(100)	☑
Prijmeni_vedouci	nvarchar(100)	☑
Telefon_vedouci	bigint	☑
Email_vedouci	nvarchar(150)	☑

Tabulka Tymy

Column Name	Data Type	Allow Nulls
ID	int	☐
Jmeno	nvarchar(100)	☐
Prijmeni	nvarchar(100)	☐
Prezdivka	nvarchar(100)	☑
Datum_narozeni	date	☑
Soutezni_kategorie_ID	int	☐
Okres_ID	int	☑
Hasicsky_sbor_ID	int	☑

Tabulka Zavodnici

Column Name	Data Type	Allow Nulls
ID	int	☐
Nazev	nvarchar(50)	☐

Tabulka Soutezni_kategorie

B.5 Tabulky *Kraje, Okresy, Obce*

Column Name	Data Type	Allow Nulls
ID	int	☐
Nazev	nvarchar(50)	☐
Zkratka	nvarchar(10)	☑

Tabulka Kraje

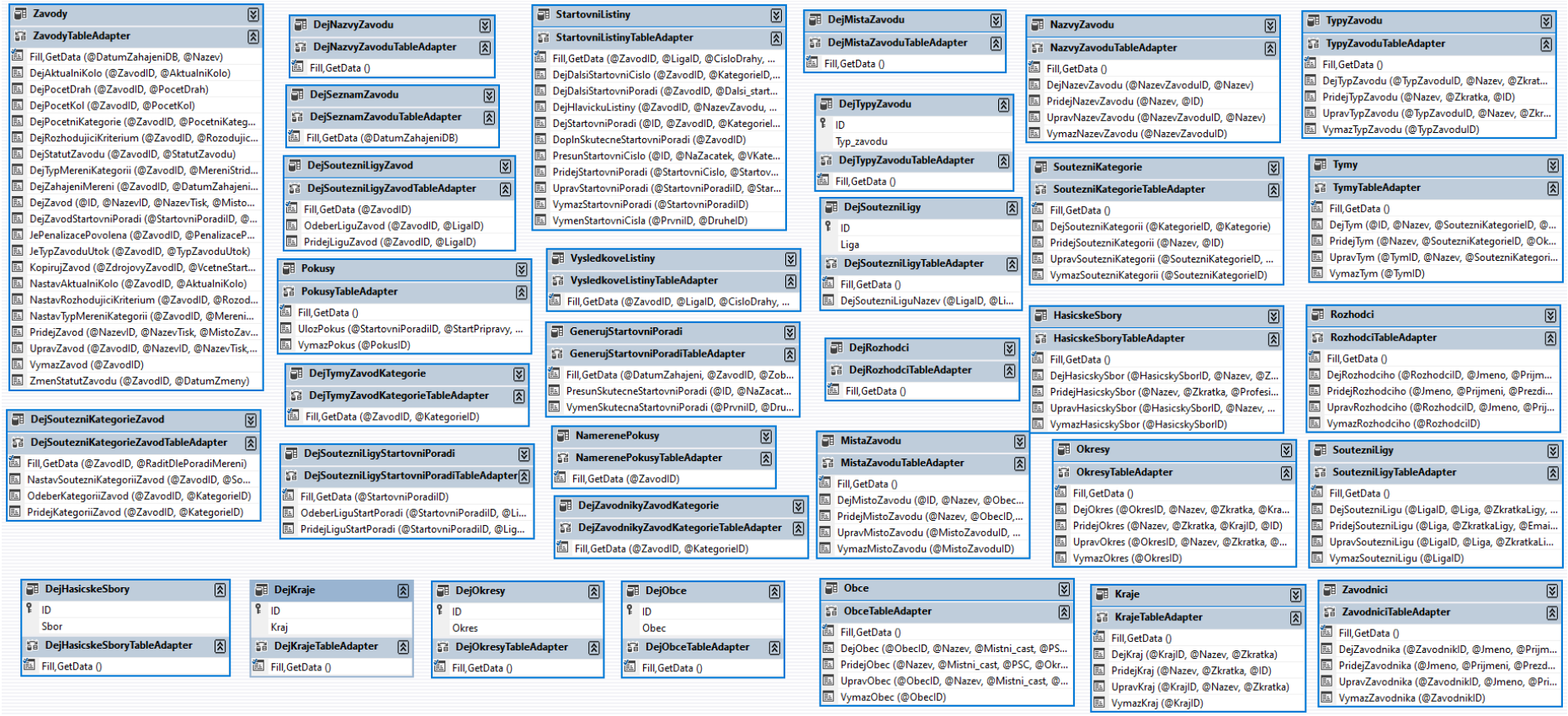
Column Name	Data Type	Allow Nulls
ID	int	☐
Nazev	nvarchar(100)	☐
Zkratka	nvarchar(10)	☑
Kraj_ID	int	☐

Tabulka Okresy

Column Name	Data Type	Allow Nulls
ID	int	☐
Nazev	nvarchar(100)	☐
Mistni_cast	nvarchar(100)	☑
PSC	int	☑
Okres_ID	int	☐

Tabulka Obce

PŘÍLOHA C – GRAFICKÝ NÁVRH DATOVÉHO ZDROJE ODPOJENÉ VRSTVY ADO.NET



PŘÍLOHA D – DIGITÁLNÍ PŘÍLOHA

Přiložené DVD obsahuje:

- Bakalářskou práci ve formátu *.pdf*
- Aplikaci Firesport Timekeeper včetně zdrojových kódů
- Binární soubor *.bin* se zálohou měření miniaplikace Časomíra
- Zálohu databáze pro zpracování výsledků ve formátu *.bacpac*
- ER diagram databáze
- Obrázky s návrhy databázových tabulek
- Soubory s implementovanými SQL kódy
- Aplikaci LV Tester zastupující časomíru LV 4,66 a sloužící pro testování programu Firesport Timekeeper
- Zdrojové kódy aplikace LV Tester
- Stručné návody pro instalaci a konfiguraci obou aplikací a dalšího potřebného softwaru