TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

T-MC informační systém pro malé firmy na platformě SAP

Bakalářská práce

Studijní program: B2612 – Elektrotechnika a informatika
Studijní obor: 1802R022 – Informatika a logistika

Autor práce: **Matěj Haase**
Vedoucí práce: RNDr. Klára Císařová, Ph.D.

Liberec 2016

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

T-MC information system for small companies in SAP environment

Bachelor thesis

Study programme: B2612 – Electrical Engineering and Informatics
Study branch: 1802R022 – Informatics and Logistics

Author: **Matěj Haase**
Supervisor: RNDr. Klára Císařová, Ph.D.

Liberec 2016

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií
Akademický rok: **2015/2016**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Matěj Haase**
Osobní číslo:	**M12000022**
Studijní program:	**B2612 Elektrotechnika a informatika**
Studijní obor:	**Informatika a logistika**
Název tématu:	**T-MC informační systém pro malé firmy na platformě SAP**
Zadávající katedra:	**Ústav mechatroniky a technické informatiky**

Zásady pro vypracování:

1. Seznamte se s vznikem a historií SAP, jeho programovacím jazykem ABAP.
2. Analyzujte firemní požadavky a navrhněte datový model s důrazem na napojení na standardní SAP datový model.
3. Popište objektový model již existujícího T-MC-IS frameworku.
4. Realizujte dílčí částí datového modelu pomocí T-MC-IS frameworku.
5. Zhodnocení přínosů/nedostatků uživatelského rozhranní T-MC-IS proti standardní údržbě dat.

Rozsah grafických prací: **dle potřeby dokumentace**

Rozsah pracovní zprávy: **cca 30–40 stran**

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

[1] **Karl-Heinz Kühnhauser: ABAP výukový kurz, Computer Press, a.s. , 2009**

[2] **André Maassen, Andreas Gadatsch, Detlev Frick, Markus Schoenen: SAP R/3 Kompletní průvodce, Computer Press, a.s., 2007**

[3] **George W. Anderson: Naučte se SAP za 24 hodin, Computer Press, a.s., 2012**

[4] **KŘIVÁK, V. Návrh a realizace komplexního monitorovacího systému pro podnikový IS na bázi SAP. Praha, 2008. Diplomová práce. ČVUT.**

[5] **KÜHNHAUSER, K. - H. ABAP: Výukový kurz. 1. vydání. 2009. ISBN 978-80-251-2117-7.**

Vedoucí bakalářské práce: **RNDr. Klára Císařová, Ph.D.**
Ústav mechatroniky a technické informatiky

Konzultant bakalářské práce: **Ing. Pavel Kaiser**
T-MC66, s.r.o., Praha

Datum zadání bakalářské práce: **10. října 2015**

Termín odevzdání bakalářské práce: **16. května 2016**

prof. Ing. Václav Kopecký, CSc.
děkan

Technická univerzita v Liberci - 14 -

L.S.

doc. Ing. Milan Kolář, CSc.
vedoucí ústavu

V Liberci dne 10. října 2015

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum: 15.5.2016

Podpis:

Poděkování

Na tomto místě bych rád poděkoval všem, kteří se jakýmkoliv způsobem podíleli na vývoji této bakalářské práce.

V první řadě bych rád poděkoval svému konzultantovi bakalářské práce Ing. Pavlu Kaiserovi a mému zaměstnavateli Ing. Lukáši Sýkorovi za jejich velkou pomoc, cenné rady a obrovskou trpělivost, kterou se mnou měli.

Dále děkuji vedoucí práce RNDr. Kláře Císařové, Ph.D. za skvělé vedení a další cenné rady, které jsem uplatnil při psaní této bakalářské práce.

A závěrem moc děkuji rodině a kamarádům, kteří při mně po celou dobu stáli, věřili mi a zpříjemňovali mi prostředí po celou dobu studia.

Abstrakt

Hlavním tématem této bakalářské práce je detailní seznámení podnikovým informačním systém od německé společnosti SAP (Systems - Applications - Products in data processing) a vytvoření vlastní aplikace pro společnost T-MC66, s.r.o.

První část této práce se týká společnosti SAP, produktů, která vytvořila a architektury jak z hlediska technického, tak z hlediska podnikových procesů.

V druhé části je popsáno vývojové prostředí MySAP Business Suite od společnosti SAP a jimi vytvořený programovací jazyk ABAP (Advanced Business Application Programming) s ukázkami kódu, ve kterých se objevují základní příkazy tohoto programovacího jazyka.

Další část je věnována speciální části prostředí SAP a tím je vývoj vlastních aplikací, který je využíván v případě specifických požadavků zákazníka, či v případech, že nabízené standardy obsažené v základu systému nejsou dostatečné. Jednou z takovýchto zákaznických aplikací je vlastní framework vyvinutý firmou T-MC66, s.r.o., která je současně zadavatelem této bakalářské práce.

Poslední částí této práce je tvorba aplikace pro evidenci a zpracování firemních dat ve firmě T-MC66 s.r.o (např. vykazování odpracovaných hodin na určité zakázce) s návazností na zmíněný framework a standardní SAP objekty.

Klíčová slova

SAP, programový modul, ABAP, podnikový informační systém, evidence dat

Abstract

The main topic of this bachelor thesis is to explore and study ERP (Enterprise Resource Planning) system from German company SAP SE (Systems – Applications – Products in data proccessing) and create own application for T-MC66, s.r.o company.

In first part can be found information about SAP company, their products and architecture from technical perspective and from perspective of the company processes.

In the second part is described development environment of MySAP Business Suite from SAP and it's programming language ABAP (Advanced Business Application Programming) with code examples.

The next part is dedicated to a special part of SAP enviroment which is development of own applications, that is used in case of specific company requests where the basic SAP elements are not sufficient. Very own framework is one of this application developed by T-MC66 s.r.o, which is also contracting authority of this work.

The last part of this work is the creation of an application for data recording and processing in T-MC66, s.r.o. company (eg. Reporting of worked hours on a contract) with relation to mentioned framework and SAP standard objects.

Keywords

SAP, program module, ABAP, enterprise resourse planning, data evidence

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek

Zkratka	Popis
ABAP	Advanced bussines application program
ALV	ABAP list viewer
B2B	Bussines to bussines
CRM	Customer relationship manager
ERP	Enterprise resource planning
GUI	Graphical user interface
MVC	Model-View-Controller
SAP	Systems – Applications – Products in data proccessing
SE11	Transakce pro tvorbu tabulek a struktur
SE80	Transakce Object Navigator

Úvod

V dnešní době každá společnost hledá různé prostředky a možnosti, díky kterým by mohla zredukovat své celkové náklady. Jedna z možností jak docílit nižších nákladů je snížit celkový čas procesů, které jsou nedílnou součástí každé společnosti a právě tímto tématem se zabývá i společnost SAP, která se za pomoci různých softwarových řešení snaží snížit celkový čas, zefektivnit a optimalizovat jednotlivých firemních procesů.

Tato práce se zabývá konkrétním řešením pro jednoduchou evidencí a zpracování firemních dat u firmy T-MC66, s.r.o., kde se před zavedením nového řešení prováděli veškeré evidence buď v excelovských tabulkách nebo tradičních dokumentech typu word a kde veškerá komunikace byla řešena pomocí úložišť dokumentů (cloudů), které byly doplňovány ústní nebo e-mailovou komunikací.

Nové řešení by mělo firmě T-MC66, s.r.o. pomoci ucelit tyto evidence do jednotného optimálního systému a díky schvalovacím procesům tak i ulehčit celkovou komunikaci mezi všemi členy firemní organizační struktury a díky tomu zkrátit čas těchto procesů a tedy i celkových nákladů.

1 Úvodní studie

1.1 Podnikový informační systém

Podnikový informační systém, označovaný také jako ERP (Enterprise resource planning), je systém, díky kterému je možné řídit a integrovat většinu podnikových procesů jako jsou: plánování, zásobování, nákup, prodej, marketing, finance atd. I přes to, že dnes se bere jako samozřejmost je podnikový informační systém mimořádně složitý komplex hardwaru a softwaru (viz kapitola 2 Architektura SAP). ERP napomáhá podnikům řídit vlastní procesy na všech úrovních jejich architektury a podnik je tudíž schopný komunikovat a sdílet informace se všemi ostatními pracovními skupinami v rámci celé organizace přičemž je zde i možné určit oprávnění a přístupy pro každou pracovní skupinu zvlášť. Právě společnost SAP byla první firmou, která s ERP softwarem přišla na trh. [1][2]

1.2 O společnosti SAP SE

Společnost SAP SE je v současné době z hlediska tržní kapitalizace třetím největším nezávislým výrobcem software. Sídlí v německém Walldorfu a specializuje se především na vývoj ERP systému pro B2B (business to business) segment. V současné době je SAP největším dodavatelem softwaru v oblasti podnikových aplikací. Velký podíl na rychlém vzestupu společnosti SAP však také měli další velké společnosti jako například IBM, Microsoft či Oracle a to hlavně díky kompatibilitě a spolupráci těchto firem. Například Oracle je největším dodavatelem databází a společnost Microsoft dodává operační systémy pro datová centra, zároveň je SAP nejčastěji provozován právě na těchto systémech. [2][3][4]

1.3 Historie společnosti SAP

Společnost SAP (Systeme, Anwendungen, Produkte in der Datenverarbeitung) byla založena roku 1972 v Mannheimu v Německu, a to skupinou bývalých zaměstnanců IBM (Dietmar Hopp, Hans-Werner Hector, Hasso Plattner, Klaus Tschira a Claus Wellenreuther) jejichž cílem bylo vyvinout softwarový balík, který by obsahoval co možná nejvíce podnikových funkcí.

Tomuto cíli se přiblížili po vydání finanční aplikace RF a systémem pro správu materiálu RM. Začátkem roku 1973 získal SAP svého prvního zákazníka a to chemickou společnost ICI. V roce 1977 se společnost přesouvá do současného sídla ve městě Walldorf a vůbec poprvé získává zákazníky za hranicemi Německa. V 90. letech již produkty z dílny SAP využívá skoro 80% největších německých společností. Zlomovým okamžikem bylo uvedení předchůdce ERP systému a to produktu R/3 v polovině roku 1992, který zahájil expanzi do malých a středních podniků.

Během několika dalších měsíců se SAP stal největším softwarovým výrobcem v Německu a 7. největším na světě a roku 1993 prodeje překonávají hranici 1 miliardy marek. Ten samý rok společnost SAP rozrůstá i do České republiky. V dnešní době má SAP okolo 300tis. zákazníků ve 190 zemích a je jednou z největších softwarových firem v oblasti podnikových aplikací. [2][4][5]

1.4 Produkty společnosti SAP

Asi nejznámějším produktem je e-business integrační software MySAP (označovaný také jako SAP Business Suite) poskládaný z příslušných aplikačních komponent zvaných SAP Components. Tyto aplikační komponenty obsahující více než 200 předdefinovaných šablon rolí, které poskytují uživatelům přístup k aplikacím a zdrojům a umožňují snadnější komunikaci a předávání informací v rámci organizace. Mezi nejznámější komponenty patří:

- SAP ERP (Enterprise Resource Planning)
- SAP CRM (Customer Relationship Management)
- SAP SCM (Supply Chain Management)
- SAP SRM (Supplier Relationship Management)
- SAP SEM (Strategic Enterprise Management)
- a další

Další produkt, který je třeba zmínit je SAP NetWeaver. SAP NetWaver je primární technologickou platformou pro většinu aplikací společnosti SAP a to i balíku SAP Business Suite s využitím webového rozhraní. Charakteristický je především svými čtyřmi integračními úrovněmi: integrace osob, integrace informací, integrace procesů a aplikační platforma, která je kompatibilní s hlavními technologickými platformami současnosti, jako jsou Java 2 Enterprise Edition (J2EE); Microsoft .NET; nebo IBM WebSphere. [2][8]

1.5 SAP Česká republika

SAP ČR je dceřiná společnost SAP SE působící v České republice již od roku 1992. Díky znalosti lokálního trhu se českému zastoupení podařilo navázat mimořádně úspěšnou komunikaci se zdejšími podniky a organizacemi. Výsledkem je více než 1 200 českých zákazníků, z nichž velká většina patří ke špičce českého hospodářství a státní administrativy.

SAP ČR v současné době zaměstnává 272 pracovníků ve dvou pobočkách, kteří se kromě obvyklé obchodní a marketingové činnosti věnují také dalším vysoce kvalifikovaným činnostem jako výzkum a vývoj, poradenské a implementační služby, školení a podpora zákazníků. V Praze se nachází regionální zastoupení SAP ČR, Brněnská pobočka funguje jako vývojové a lokalizační středisko pro celou střední a východní Evropu. V Praze se také nachází třetí a současně největší zastoupení SAP v České republice - SAP Business Services Centre Europe. Centrum sdílených služeb má téměř 700 zaměstnanců, kteří poskytují podporu v oblasti financí a lidských zdrojů pro pobočky v Evropě a Africe. [6]

1.5.1 Nasazení SAP v ČR

V následující tabulce je uveden výpis významných společností na území české republiky, které využívají systém SAP.

Tabulka 1: Firmy využívající SAP v ČR

Společnost	**Odvětví**
Škoda auto, a.s.	Automotive
Witte Automotive, s.r.o.	Automotive
Johnson Controls, k.s.	Automotive
Magna Automotive, s.r.o.	Automotive
ČEZ, a.s.	Energetika
RWE, a.s.	Energetika
E.on, a.s.	Energetika
T-Mobile, a.s.	Komunikace
MobilKom, a.s.	Komunikace
DHL Express, s.r.o.	Služby
Česká pošta, s.p.	Služby
Agrofert Holding, a.s.	Potravinářství
Danone, a.s.	Potravinářství
Preciosa, a.s.	Bižuterie
Okay, s.r.o.	Elektronika
BD Senesors, s.r.o.	Elektronika
Deloitte	Finančnictví
ICZ, a.s.	Software

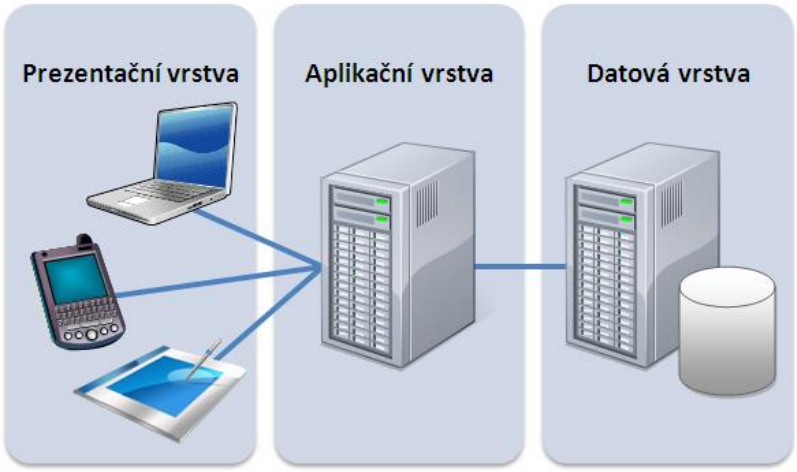
2 Architektura SAP

2.1 Technická architektura

Technická architektura systému SAP je v základu rozdělena na 3 vrstvy a to na:

- prezentační vrstvu
- aplikační vrstvu
- databázová vrstvu

Toto rozdělení vytváří třívrstvou architekturu klient-server, která je založena na principu distribuce práce (viz obr. 1). [7]



Obrázek 1: Architektura klient-server (zdroj: https://managementmania.com/cs)

Prezentační vrstva napomáhá komunikaci mezi uživatelem a počítačem a to za pomocí dialogových oken. Toto je umožněno díky programovému balíku pro grafické uživatelské prostředí SAP GUI. SAP GUI si můžeme představit jako jakýsi most mezi prezentačním a aplikačním serverem, po kterém je vedena veškerá komunikace.

Aplikační vrstva slouží k zpracovávání dat a to na aplikačním serveru. Tento server se stará o komunikaci s uživatelským prostředím SAP GUI a dále komunikuje s databází za pomocí systému pro správu databází. Tato vrstva je důležitá zejména pro běh programů na těchto serverech.

Databázová vrstva jak už název napovídá, obsahuje databázový server, na kterém jsou uložena veškerá data, ale i nastavení systému nebo programy. V závislosti na množství dat může být databázový server uložen i na více počítačích (toto platí i u vrstvy aplikační).

Informace jsou převzaty z literatury [7].

2.2 Architektura z pohledu využití v podniku

Mezi další možnosti jak lze systém dělit je z hlediska využití v podniku. Pro zavedení a následný provoz systému SAP nestačí mít nainstalovaný pouze jeden produktivní systém, ale vždy je nutné vytvořit i další neproduktivní systémy. Společnost SAP doporučuje využití tří základních systémů:

- Vývojový systém
- Testovací a předávací systém
- Produktivní systém

Každý z těchto systémů má své opodstatnění a cílovou skupinu uživatelů. Vývojový systém mají na starosti IT vývojáři, testovací a předávací systém klíčoví uživatelé a produktivní sytém je výslednou částí pro koncové uživatele. Veškeré aktivity customizingu či vývoje mohou být prováděny i na několika vývojových klientech (jeden systém může obsahovat i více klientů) a jejich výstup se poté přenáší pomocí takzvaných transportních příkazů.

3 Vývojové prostředí SAP

Vzhledem k tomu, že je tato bakalářská práce zaměřená zejména na vývoj, nebudu zde příliš zabíhat do detailů prostředí SAPu jako je například logování do systému nebo úvodní obrazovka, ale pokusím se v této kapitole popsat hlavně transakci SE80 neboli Object Navigator z části ABAP Workbench a nejdůležitější objekty se kterými je možné se v SAPu setkat.

3.1 ABAP Workbench – Object Navigator

Object Navigator je základním prostředím každého vývojáře, jelikož jeho menu obsahuje přehled všech dosud uživatelsky vytvořených tabulek, datových prvků atd. a dovoluje je okamžitě prohlížet, editovat, mazat či vytvářet nové. Je samozřejmostí, že toto jde také pomocí samostatných transakcí (např. transakce SE11 pro správu tabulek) avšak obrovská výhoda transakce ABAP Workbench je, že vývojář nemusí pokaždé otevřít novou transakci, ale stačí mu si daný prvek v pravém menu vybrat a ať už je to tabulka či datový prvek okamžitě se zobrazí do pravé části obrazovky, kde je možné s ním okamžitě pracovat. Kompletní popis prostředí transakce Object Navigator je možné nalézt v příloze číslo 1.

3.2 ABAP Objekty

Každý uživatel, ať už se jedná o vývojáře nebo koncového uživatele, musí znát pro užívání SAPu alespoň základní objekty, se kterými se dostane do styku. Mezi tyto objekty pak patří zejména pakety, databázové tabulky, datové prvky, domény, struktury, programy a transakce.

- ***Paket*** **–** svazuje související objekty určité úlohy a definuje transportní vrstvu, která určuje, zda objekty budou přeneseny či ne. Tedy pokud program chceme přenést do produkčního systému, musíme paket vyplnit odpovídající hodnotou v opačném případě zůstane objekt pouze v systému pro testování a vývoj pod lokálním paketem $TMP.

❖ ***Objekty data dictionary***

- ***Databázové tabulky*** – jsou základem celého systému SAP. Celý systém je rozdělen na programovou část a datovou část. Programová část se využívá pro přístup k veškerým datům uloženým právě v databázových tabulkách (pro představu počet databází systému SAP se pohybuje v řádech desetitisíců až statisíců). Samozřejmostí je i indexace u jednotlivých tabulek. Index dále obsahuje ukazatel na konkrétní záznam. Rozeznáváme 3 druhy databázových tabulek a to transparentní tabulky, tabulky poolu, clusterové tabulky.

  Transparentní tabulky – do těchto tabulek se ukládají zejména data aplikací a jejich fyzická reprezentace v databázi přesně odpovídá definici v nástroji ABAP Dictionary.

  Tabulky poolu a clusterové tabulky – v těchto tabulkách je uložena dokumentace, nastavení a sekvence dynamických programů. Na rozdíl od transparentních tabulek jsou však data uložena v samostatném poolu či clusteru tabulek ve fyzické databázi. Pool tabulek tvoří jediná tabulka ve fyzické databázi, které jsou identifikovány názvem a klíčovými poli. Opoti tomu cluster tabulek se skládá z více clusterových tabulek, které jsou kombinovány pomocí klíčů.

- ***Datové prvky*** – jsou vlastně záznamy z databázové tabulky. Datový prvek je elementární typ, který je popsán klíčem, zda se jedná o inkrementující hodnotu (např. index) a doménou, která pak udává jeho další vlastnosti.

- ***Domény*** – popisují datový prvek. Obsahují veškeré informace jako je např. datový typ, počet míst (případně i desetinných míst) nebo také rozsah hodnot kterých může datový prvek nabývat

- ***Databázové struktury –*** je skupina interních polí které spolu vzájemně logicky souvisejí.Typické jsou tím, že data uchovávají pouze po dobu běhu programu. Na rozdíl od databázových tabulek struktura neobsahuje ani neodráží žádnou tabulku z datového slovníku ABAP/4, nemá primární klíč a nemá přiřazené

žádně technické vlastnosti, jako jsou: třída, velikost kategorie či parametry týkající se použití bufferů.

- ***Views*** **–** údaje o aplikačním objektu jsou většinou distribuovány v několika tabulkách a právě díky views můžeme definovat pohledy, které kombinují tyto data. Views se mohou být dále použity v ABAP programech pro výběr dat.

❖ ***Knihovna tříd***

- ***Třídy*** **–** jsou šablony pro objekt. Je to jakýsi soubor instrukcí pro stavbu objektu, jelikož vlastnosti objektu jsou dány komponentami třídy, které popisují jeho chování.

- ***Rozhraní*** **–** jsou nezávislé struktury, které rozšiřují rozsah tříd, přidávají vlastní komponenty do veřejné části a umožňují uživateli adresovat různé třídy z jediného bodu.

❖ ***Programy***

- ***Spustitelné programy*** **–** mohou obsahovat všechny procesní bloky s vyjímkou funkčních modulů a umožňují práci s databázovými tabulkami. Veškeré spustitelné programy začínají příkazem REPORT *název programu*. Více viz 3.3.1 Struktura programů.

- ***Programy typu Include*** **–** slouží výhradně pro modularizaci zdrojového kódu. Tento typ programů si můžeme představit jako funkci, kde se naprogramuje a dále využije jiným programem. Může to být například deklarační část nebo třeba i dlouhý kus kódu.

❖ ***Skupina funkcí***

- ***Funkční moduly*** **–** mohou být označovány též jako skupiny funkcí. Tyto skupiny funkcí dovolují uživateli vytvořit „globální“ funkci v centrální knihovně, která může být následně volána z kteréhokoliv programu ABAP.

- ***Transakce*** **–** slouží ke spouštění programů za pomocí kódu transakce. SAP obsahuje desítky kódů pro základní SAP programy (např. pro ABAP Workbench je kód transakce SE80), ale je zde i možné vytvořit si kód transakce pro uživatelské programy. Tento kód má jmenné omezení a není nutné ho definovat.

- ***Třída zpráv*** **–** se využívá pro komunikaci programu s uživatelem, zejména pak pro chybová hlášení. Třída zpráv se skládá z jednotlivých řádků, kde každý řádek obsahuje jednu zprávu, která obsahuje tříznakové ID přes které je volána.

Informace jsou převzaty z [4][7].

3.3 Základy vývoje aplikačních programů

V programovacím jazyce ABAP rozpoznáváme 2 druhy aplikačních programů – reporty a dynamické programy.

- ***Reporty*** jsou programy jejichž výstupem jsou seznamy, které jsou vygenerovány na základě vstupních dat, které uživatel zadává do tzv. výběrové obrazovky (Selection screen). Základní vlastností reportu je, že na rozdíl od dynamických programů uživatel pouze zadá vstupní data a dostane výstup.

- ***Dynamické programy,*** také známé jako ***dynpra,*** jsou programy složené z více obrazovek a naopak od reportů, kde do průběhu programu uživatel nemohl zasáhnout, je zde celkové chování programu uživatelem značně ovlivněno.

3.3.1 Struktura programů

Každý spustitelný program jazyka ABAP se dělí na dvě části a to:

deklarační část – zde se definují veškeré tabulky, struktury, pole atd. Samozřejmostí je deklarace globálních dat (tzn. dat využívaných v celém programu) a lokálních dat (dostupné pouze v dané proceduře), ale i deklarace výběrových obrazovek se všemi parametry a definicemi.

bloky zpracování – představují jednotlivé úlohy volané dle specifických pravidel. Obsahují dialogové obrazovky (např. volání výběrových obrazovek), bloky událostí a

procedury. Tradičně blok zpracování začíná a končí událostí. Začátek takovéto události je uvozen klíčovými slovy např. START-OF-SELECTION či AT SELECTION-SCREEN přičemž pořadí událostí je pevně dané tzn. že program bude postupně vyhledávat jednotlivé bloky událostí a to v pořadí:

1) LOAD-OF-PROGRAM – provede se okamžitě po spuštění programu
2) INITIALIZATION – v této události se provádí deklarace dat pro výběrové obrazovky nebo tuto obrazovku upravují
3) AT-SELECTION-SCREEN – slouží zejména pro složitější kontroly vstupu. Například po potvrzení výběrové obrazovky provede kontrolu, zda uživatel zadal platné hodnoty a případně vyvolá chybné hlášení a donutí uživatele zadat hodnoty znovu
4) START-OF-SELECTION – pokud jsou všechna data „v pořádku“, systém se dostane k události START-OF-SELECTION, kde už by měl být výpis reportu, který se uživateli na základě zadaných dat zobrazí [7]

3.3.2 Deklarace tabulek, proměnných a ukazatelů

V první řadě se vždy deklarují tabulky, se kterými chceme pracovat a to pomocí klíčového slova TABLES a teprve následně jednotlivé proměnné nebo ukazatele. Deklarace proměnných či polí je uvozena klíčovým slovem DATA, u ukazatelů je to FIELD-SYMBOL. Proměnné mají jmenná omezení to znamená, že není možné použít jakékoliv jméno jaké nás napadne (např. název musí vždy začínat písmenem a může obsahovat maximálně 30 znaků atd.).

Při deklaraci proměnné je nutné uvést také typ – TYPE. Typ udává, zda se jedná o hodnotu, textový řetězec či jiný typ a pokud je proměnná dále použita ke komunikaci s uživatelem za pomocí výběrové obrazovky, je pak tímto dáno i omezení tzn. jaké hodnoty uživatel může do jednotlivých polí zapsat.

Ukázka kódu

```
1  *REPORT
2  REPORT  Z_PRVNI_REPORT.
3
4  *DEFINICE TABULEK
5  TABLES ztabulka.
6
7  *DEKLARACE DAT
8  *deklarace proměnné uzivatel s typem c (char - textový řetězec)
9  DATA uzivatel TYPE c.
10
11 *deklarace proměnné id s typem i (integer - hodnota) s hodnotou (VALUE) jedna
12 DATA id TYPE i VALUE 1.
13
14 *deklarace proměnné jmeno, která bude mít stejné (LIKE) parametry jako proměnná uživatel
15 DATA jmeno LIKE uzivatel.
16
```

Zdrojový kód 1: Deklarace dat

Poznámka: Jak si můžete všimnout tak jak název reportu tak i tabulky začíná písmenem Z (může to být i písmeno Y). Toto je opět z důvodu jmenných konvencí, aby se rozlišilo zda se jedná o standardní objekt systému SAP nebo o objekt zákazníka.

3.3.3 Deklarace a implementace tříd a rozhraní

Deklarace třídy je uvozena klíčovým slovem CLASS a rozhraní klíčovým slovem INTERFACE.

Třídy a rozhraní tvoří základ objektově orientované části programovacího jazyka ABAP a mohou obsahovat jednotlivé komponenty jako datové typy a atributy, metody nebo události. Tyto komponenty se pak deklarují přímo v určité třídě či rozhraní. Hlavní rozdíl mezi třídou a rozhraním je, že třídy obsahují jak definiční část, tak i implementační část, zatímco rozhraní obsahují pouze definiční část a implementovány mohou být právě přes třídy. Příkladem mohou být třeba hodiny které budeme chtít rozšířit o funkci budíku. Pokud budeme mít třídu HODINY (CLASS HODINY) a my budeme chtít třídu HODINY S BUDÍKEM (CLASS HODINY_BUDIK) budeme potřebovat další tlačítka na ovládání tohoto budíku, které se nastavují právě rozhraním. Tedy pro naši třídu HODINY_BUDIK zdědíme třídu HODINY a rozšíříme ji rozhraním BUDIK. Tedy

rozhraní ve třídě HODINY_BUDIK definuje a implementuje tyto dvě metody v implementaci třídy.

3.3.4 Načítání a textový výpis dat z databáze

Pro čtení a výpis dat je nutno znát další dva příkazy a to SELECT a WRITE. Příkaz SELECT kontroluje řádek po řádku (záznam po záznamu) definovanou tabulku a za pomocí dalších specifikací vybere všechny odpovídající záznamy (více viz 3.3.6 Cykly a struktury). Příkaz WRITE pak tyto záznamy vypíše. Níže ve zdrojovém kódu 2 je možné si prohlédnout jejich obdoby.

Ukázka kódu

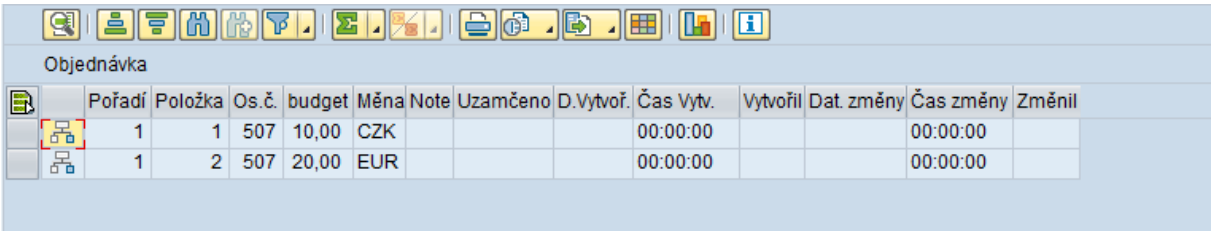
```
1  *REPORT
2  REPORT Z_PRVNI_REPORT.
3
4  *DEFINICE TABULEK
5  TABLES ztabulka.
6
7  *vybereme všechna data z tabulky ztabulka a vypíšeme je
8  SELECT * FROM ztabulka.
9    WRITE / ztabulka.
10 ENDSELECT.
11
12 *vybereme všechna data z tabulky ztabulka, ale vypíšeme pouze jmena
13 SELECT * FROM ztabulka.
14   WRITE / ztabulka-jmeno.
15 ENDSELECT.
16
17 *vybereme pouze data, kde datový prvek jmeno obsahuje jméno tomáš a vypíšeme je
18 SELECT * FROM ztabulka WHERE jmeno = 'tomáš'.
19   WRITE / ztabulka.
20 ENDSELECT.
21
```

Zdrojový kód 2: Načítaní dat z databáze

Poznámka: Lomítko za příkazem WRITE značí nový řádek. Díky tomu je při závěrečném výpisu reportu každý záznam na samostatném řádku.

3.3.5 Grafický výpis dat – ALV

Grafický výpis, který je řešen pomocí ALV (ABAP List Viewer), je asi nejčastěji využívaný. Je to z důvodu, že obyčejný výpis reportu je jenom seznamem námi vybraných dat, zatímco grafický výpis vyobrazuje data jako tabulku nebo strom a obsahuje mnoho užitečných funkcí, které by jinak programátor musel vytvořit. Mezi tyto funkce patří: řazení záznamů, vyhledávání, filtrování, sčítání sloupců atd.



Obrázek 2: Grafický výpis dat ALV

3.3.6 Cykly a struktury

Jak cykly (smyčky) tak struktury mají společné to, že se oba spustí pouze v případě, že je splněna podmínka. Rozdílem je pak, že struktury se po splnění podmínky provedou pouze jednou zatímco cykly mohou provádět příkazové bloky i vícekrát. Mezi cykly patří: SELECT – ENDSELECT, LOOP – ENDLOOP a WHILE – ENDWHILE a mezi struktury: IF – ENDIF, CASE - ENDCASE.

Cyklus LOOP – ENDLOOP

Smyčka LOOP – ENDLOOP je obdobná se smyčkou SELECT – ENDSELECT rozdíl je v tom, že se využívá pouze při prácí s interními tabulkami tedy tabulkami, které jsou uloženy pouze za běhu programu. Stejně jako u SELECT – ENDSELECT je i interní tabulka příkazem LOOP – ENDLOOP zpracovávána řádek po řádku.

Ukázka kódu

```
*REPORT
REPORT Z_PRVNI_REPORT.

*DEFINICE TABULEK
TABLES ztabulka.

*Definice interní tabulky
DATA: BEGIN OF intertab OCCURS 3,
ijmeno LIKE ztabulka-jmeno,
iid LIKE ztabulka-id,
idatum LIKE ztabulka-datum,
END OF report.

*výběr dat z databázové tabulky ztabulka a následný přenos dat do interní tabulky intertab
SELECT * from ztabulka.
MOVE-CORRESPONDING ztabulka to intertab.

append intertab.

clear intertab.

ENDSELECT.

*Výpis dat interní tabulky
LOOP at intertab.

WRITE : / intertab-ijmeno,
intertab-iid,
intertab-idatum.

ENDLOOP
```

Zdrojový kód 3: Práce s interními tabulkami

Cyklus DO – ENDDO

U smyčky DO – ENDDO je důležité uvést kolikrát se má provést. Toho je docíleno dodatkem n TIMES, kde n je celé číslo typu i. Pokud není uveden počet opakování nebo není vložen kód pro ukončení smyčka se bude opakovat stále dokola. Počet provedených cyklů je uložen v systémové proměnné **sy-index**. Důležitá je zde hlavně podmínka, která je čtena až po prvním projetí smyčky tedy i kdybychom chtěli uvést podmínku zda se cyklus má vlastně spustit bude přečtena až po prvním projetí celého cyklu DO – ENDDO.

Ukázka kódu

```
1   *REPORT
2   REPORT Z_PRVNI_REPORT.
3
4   *DEFINICE TABULEK
5   TABLES ztabulka.
6
7   *Příkaz který 5x vypíše systémovou proměnnou sy-index
8   DO 5 TIMES.
9     WRITE sy-index.
10  ENDDO.
11
```

Zdrojový kód 4: Smyčka DO

Poznámka: Výpisem tohoto jednoduchého programu budou hodnoty 1,2,3,4,5 jelikož při každém projetí cyklu je proměnná sy-index inkrementována o 1.

Cyklus WHILE – ENDWHILE

Cyklus WHILE – ENDWHILE je obdobný cyklu DO – ENDDO, ovšem zde už systém ověřuje podmínky na začátku cyklu a při jejich nesplnění přeskočí na konec (ENDWHILE) a pokračuje dalšími příkazy programu.

Ukázka kódu

```
1   *REPORT
2   REPORT Z_PRVNI_REPORT.
3
4   *DEFINICE TABULEK
5   TABLES ztabulka.
6
7   DATA text TYPE string VALUE 'hello world'.
8
9   *Záměna mezer v proměnné text za pomlčku
10  WHILE sy-subrc = 0.
11    REPLACE space WITH '-' INTO text.
12  ENDWHILE.
13
```

Zdrojový kód 5: Smyčka WHILE

Poznámka: Zde je využita další ze systémových proměnných a to sy-subrc. Tato proměnná uchovává hodnotu o úspěchu či neúspěchu provedení operací (0 – úspěch, 4 – neúspěch).

Struktura IF – ENDIF

Struktura IF – ENDIF je jednou z nejjednodušších struktur v jazyce ABAP. Hned za příkazem IF se určuje podmínka a za ní již jednotlivé příkazy, které se provedou po splnění dané podmínky. Ve struktuře IF – ENDIF je ještě možné uvést příkaz ELSE, který se naopak provede když podmínka splněna nebude.

Ukázka kódu

```
1   *REPORT
2   REPORT Z_PRVNI_REPORT.
3
4   *DEFINICE TABULEK
5   TABLES ztabulka.
6
7   DATA cislo TYPE integer VALUE 5.
8
9   IF cislo <> 0.
10    cislo = cislo - 1.
11    ELSE.
12      WRITE 'cislo je rovno 0'.
13  ENDIF.
14
```

Zdrojový kód 6: Struktura IF

Struktura CASE – ENDCASE

Struktura CASE – ENDCASE se od struktury IF –ENDIF hodně liší a to hlavně protože je možné určit více podmínek a na tyto podmínky vytvořit jednotlivé příkazy. Příkladem může být prostá kostka, kde si na každé číslo 1 – 6 vytvořím odpověď : padne-li 1 půjdu ven, padne-li 2 půjdu cvičit, padne-li 3 půjdu do hospody, atd.

Ukázka kódu

```
1   *REPORT
2   REPORT Z_PRVNI_REPORT.
3
4   *DEFINICE TABULEK
5   TABLES ztabulka.
6
7   DATA kostka TYPE integer.
8
9   CASE kostka.
10    WHILE 1.
11      WRITE: 'JDI VEN'.
12    WHILE 2.
13      WRITE: 'JDI CVIČIT'.
14    WHILE 3.
15      WRITE: 'JDI DO HOSPODY'.
16      .
17      .
18      .
19  ENDCASE.
20
```

Zdrojový kód 7: Struktura CASE

4 Framework firmy T-MC66, s.r.o.

Tato kapitola představuje další možnost, kterou SAP nabízí a tím je vlastní dovývoj prostředí. Toho se využívá v případě, kdy základní standardní funkcionality, které SAP obsahuje nestačí specifickým podmínkám zákazníka.

Toto je případ firmy T-MC66, s.r.o. a jelikož se praktická část této práce odehrává právě v prostředí, které tato firma vyvinula, je dobré si ho představit a podívat se na hlavní funkcionality, kterými disponuje a také na to, co by vše měl v budoucnu umět.

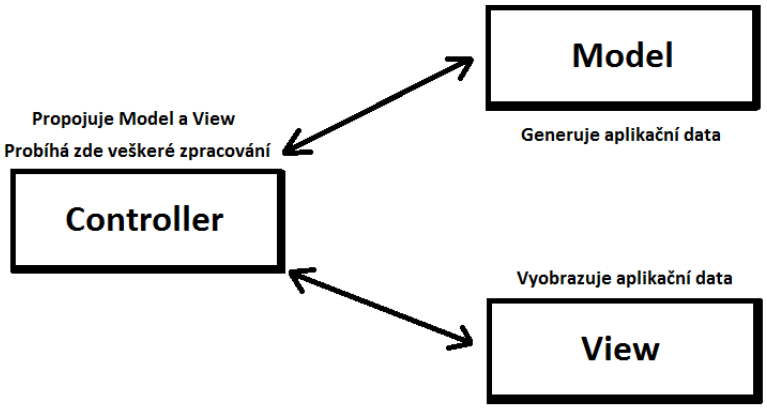
4.1 Obecný popis

Základem tohoto frameworku je jednoduchý firemní informační systém pro správu potřebných evidencí, který je vysoce modulární a postavený na obecném jádru s důrazem na možnost jeho nastavitelnosti a znovupoužitelnosti ze strany zákazníka, a to bez nutnosti zásahu dodavatele.

Tento framework by měl následně obsahovat veškerou evidenci a zpracování firemních dat, mezi které patří veškerá data o zaměstnancích jako docházka, kniha jízd, úkolovník a výkazy práce. Přičemž výkazy práce (dále jen TimeSheet) budou realizovány v praktické části této bakalářské práce.

4.2 Architektura IS

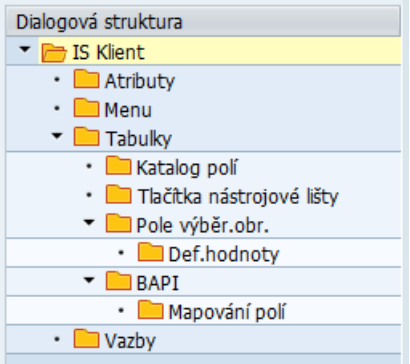
Architektura je založena na návrhovém vzoru MVC (Model-View-Controller) s plně objektovým přístupem. MVC architektura rozděluje datový model, uživatelské prostředí a řídící logiku do tří nezávislých komponent. Model obsahuje data, se kterými aplikace pracuje. View vyobrazuje tyto informace uživateli, který s nimi může dále manipulovat a controller pak na tyto podněty zajišťuje úpravy zmíněných dat (viz obr. 3).



Obrázek 3: MVC Architektura

4.3 Nastavení aplikace

Nastavení aplikace je realizováno v samostatné oddělené části (tedy ve vlastní transakci). Lze v něm nastavit jak vhled, menu, vazby, ale i celkové chování aplikace.

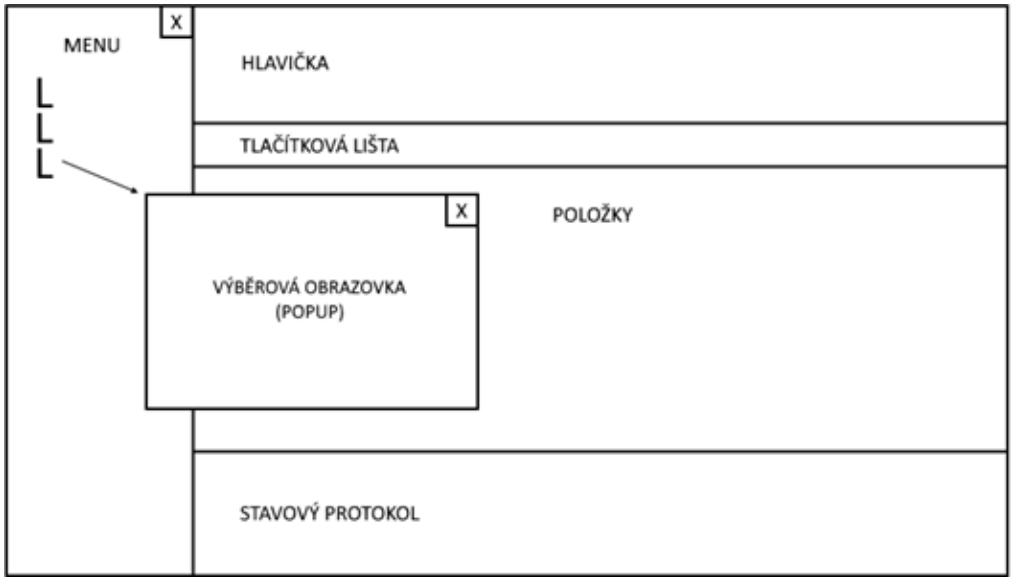


Obrázek 4: Menu nastavení

Na obr. 4 je možné vidět hierarchické rozložení menu, kde je v závislosti na klientovi možné customizovat atributy (technické parametry aplikace), menu (nastavení pozic, uzlů, ikon atd.), tabulky (každá tabulka může být nastavena rozdílně) a vazby (nastavení vazeb mezi tabulkami pro přechody mezi daty).

4.4 Vzhled aplikace

Základní rozložení hlavní obrazovky aplikace:



Obrázek 5: Framework - grafické rozložení

Menu obsahuje jednotlivé tabulky, se kterými lze pracovat.

Výběrová obrazovka - pop-up obrazovka ve které se po výběru tabulky z menu mohou definovat omezující kritéria pro výběr dat.

Hlavička je jednořádková ALV tabulka, která vyobrazuje název a typ dat, se kterými se momentálně pracuje.

Tlačítková lišta obsahuje tlačítka pro operace jako třídění dat, výmaz řádku, vytvoření nového řádku, export do excelu atd.

Položky obsahují vybraná data, která jsou opět vypisována graficky, tedy ALV tabulkou.

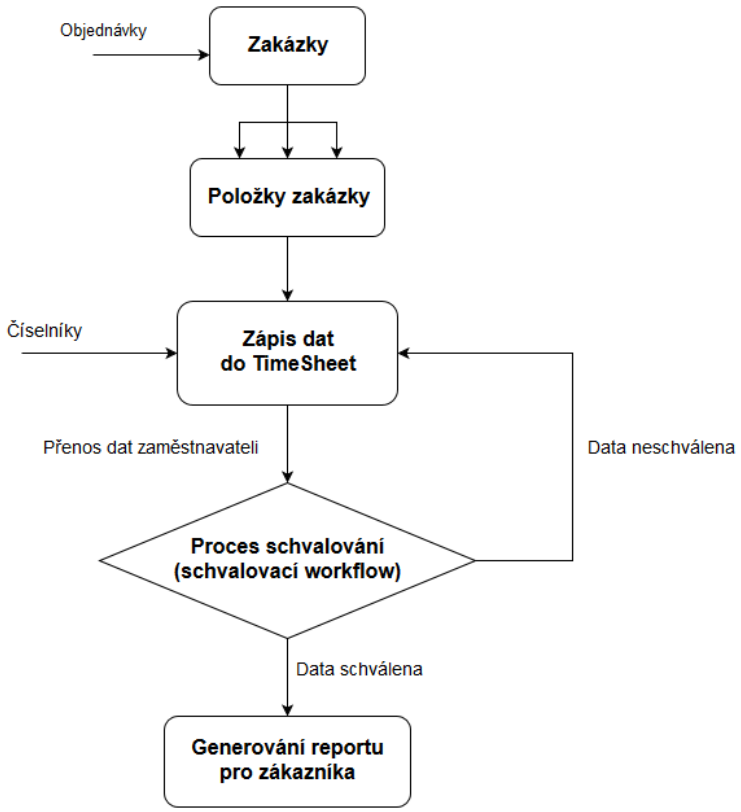
Stavový protokol zobrazuje hlášení zda operace proběhla v pořádku, případně jiná varování či chyby.

5 Specifikace požadavků, návrh a realizace

V této kapitole se podíváme na jednotlivé specifikace firmy T-MC66, s.r.o., které by má práce měla obsahovat a splňovat. A samozřejmě pak na samotný návrh a realizaci řešení.

5.1 Specifikace požadavků firmy T-MC66, s.r.o. – procesní model

Protože veškerá dosavadní evidence dat byla uložena pouze v různých „Excelových“ tabulkách a současně firma T-MC66, s.r.o. samotná se systémem SAP již pár let zabývá, rozhodla se, že by SAP mohla pro usnadnění a urychlení firemních procesů implementovat pro své interní potřeby. Úkolem bylo vytvořit a realizovat datový model pro evidenci výkazů práce, kde bude možné evidovat výkazy odpracované činnosti, schvalování a vracení záznamů (např. s využitím workflow) a plánování vytížení zaměstnanců v budoucnu. Prvním krokem bylo dohodnout se zadavatelem na struktuře a procesním modelu a na základě dohodnutých informací dále navrhnout optimální datový model, který bude realizován.



Obrázek 6: Procesní model

Struktura je řešena jednoduše s hlavním důrazem na komunikaci mezi zaměstnancem a zaměstnavatelem, a to díky procesu schvalování, který je řešen pomocí jednoduchého jednoznakového sloupce, kde jednotlivými písmeny rozlišíme, zda se jedná o nový zápis, odmítnutý zápis (neschválený) nebo správný zápis (viz tab. 2). Schválený výkaz (zápis) je dále možné odeslat zákazníkovi, buď formou reportu nebo generováním PDF souboru a ve struktuře, kterou vyžaduje zákazník.

Tabulka 2: Schvalovací workflow - identifikátory

Status	Identifikátor
Nový	N
Odmítnutý	O
Přepracovaný	P
Schválený	S

5.2 Návrh datového modelu

Samotný návrh byl nejsložitějším úkolem celé práce, a to z důvodu, že bylo nutné vzít v úvahu všechny požadavky, které firma měla, ale i z důvodu návaznosti na systém SAP. Bylo nutné přesně určit které prvky mají jednotlivé tabulky obsahovat a za pomoci externích klíčů poté mezi nimi určit návaznosti a kardinality vztahů. Návrh je řešen tak, aby bylo možné pracovat s daty co možná nejrychleji a nejefektivněji. Toho je docíleno tabulkami číselníků obsahující již předdefinovaná data o místě, aktivitě a statusu záznamu. Dále do TimeSheetu vstupuje tabulka zakázek, která přejímá informace z tabulky objednávek (viz tab. 3).

Tabulka 3: Tabulka zakázek

Pole	Klíčové pole	Datový prvek	Popis
MANDT	X	MANDT	Klient
ORDERNUMBER	X	ZCAD_ORDER	Číslo zakázky
ORDERITEM	X	ZCAD_ORDERITM	Položka zakázky
BEGDA		BEGDA	Začátek platnosti
ENDDA		ENDDA	Konec platnosti
PERNR		PERSNO	Osobní číslo
BUDGET		ZCAD_BUDGET	Rozpočet na položku
NOTE		ZCAD_NOTE	Poznámka
LOCKED		ZCAD_LOCK	Uzamčeno
CREDAT		ZCAD_CREDAT	Datum vytvoření
CRETIM		ZCAD_CRETIM	Čas vytvoření
CREUS		ZCAD_CREUS	Vytvořil
CHDAT		ZCAD_CHDAT	Datum změny
CHTIM		ZCAD_CHTIM	Čas změny
CHUS		ZCAD_CHUS	Změnil

Tabulka zakázek obsahuje kompletní informace o zakázce, tedy číslo odběratele (zákazníka), dobu platnosti zakázky nebo také rozpočet. Tato tabulka je dále rozšířena tabulkou hlavičky zakázky (viz tab. 4) a to pro lepší přehlednost zakázek, protože díky této tabulce je možné vybírat záznamy dle vlastních specifikací (např. možnost zobrazit pouze zakázky od určitého zadavatele nebo koncového zákazníka).

Tabulka 4: Hlavička zakázky

Pole	Klíčové pole	Datový prvek	Popis
MANDT	X	MANDT	Klient
ORDERNUMBER	X	ZCAD_ORDER	Číslo zakázky
BEGDA	X	BEGDA	Začátek platnosti
ENDDA	X	ENDDA	Konec platnosti
KUNNR		KUNNR	Číslo odběratele
BUDGET		ZCAD_BUDGET	Rozpočet na položku
NOTE		ZCAD_NOTE	Poznámka
CHIEV		PERSNO	Vedoucí zakázky
DEPARTMENT		ZCAD_ODD	Oddělení
TYPEOFENTRY		ZCAD_TYPEOFENTRY	Typ požadavku
NUMBEROFENTRY		ZCAD_NUMOFENTRY	Číslo požadavku
CREDAT		ZCAD_CREDAT	Datum vytvoření
CRETIM		ZCAD_CRETIM	Čas vytvoření
CREUS		ZCAD_CREUS	Vytvořil
CHDAT		ZCAD_CHDAT	Datum změny
CHTIM		ZCAD_CHTIM	Čas změny
CHUS		ZCAD_CHUS	Změnil

Další tabulkou datového modelu je tabulka objednávek (viz tab. 4), která udržuje data o interních nebo externích objednávkách, ze kterých se následně tvoří jednotlivé zakázky a jejich položky.

Tabulka 5: Tabulka objednávek

Pole	Klíčové pole	Datový prvek	Popis
MANDT	X	MANDT	Klient
PONUMBER	X	ZCAD_PO	Číslo nákupního dokladu
KUNNR		KUNNR	Číslo odběratele
PRICEMJO		ZCAD_PRICEMJO	Cílová cena za jednotku
EBELP		EBELP	Číslo položky nákupního dokladu
VBELN		VBELN	Prodejní doklad
POSNR		POSNR	Položka prodejního dokladu
XREF		ZCAD_REF	Číslo refenčního dokladu

Tedy pro upřesnění, rozdíl mezi zakázkou a objednávkou je takový, že nejprve vzniká objednávka (požadavek) – interní či externí (od zákazníka) a následně na jejím základě vzniká samotná zakázka.

Poslední a nejdůležitější tabulkou je tabulka TimeSheet (viz Tabulka 6), kde se budou evidovat výkazy jednotlivých členů firmy. V této tabulce uživatel vyplní čas strávený na vybrané položce zakázky a s doplňujícími informacemi o místě, aktivitě atd. odešle tyto informace svému zaměstnavateli ke schválení.

Tabulka 6: TimeSheet

Pole	Klíčové pole	Datový prvek	Popis
MANDT	X	MANDT	Klient
IDX	X	ZCAD_IDX	ID záznamu
PERNR	X	PERSNO	Osobní číslo
DATEOFENTRY		ZCAD_DATE	Datum záznamu
KUNNR		KUNNR	Číslo odběratele
ORDERNUMBER		ZCAD_ORDER	Číslo zakázky
ORDERITEM		ZCAD_ORDERITM	Položka zakázky
ACTIVITY		ZCAD_ACTIVITY	Druh činnosti
DESCR		ZCAD_DESCR	Popis činnosti
HRS		ZCAD_HOURS	Počet strávených hodin
PLACE		ZCAD_PLACE	Místo činnosti
INVOICED		ZCAD_INV	Fakturováno
STATUS		ZCAD_STAT	Stav záznamu
CREDAT		ZCAD_CREDAT	Datum vytvoření
CRETIM		ZCAD_CRETIM	Čas vytvoření
CREUS		ZCAD_CREUS	Vytvořil
CHDAT		ZCAD_CHDAT	Datum změny
CHTIM		ZCAD_CHTIM	Čas změny
CHUS		ZCAD_CHUS	Změnil

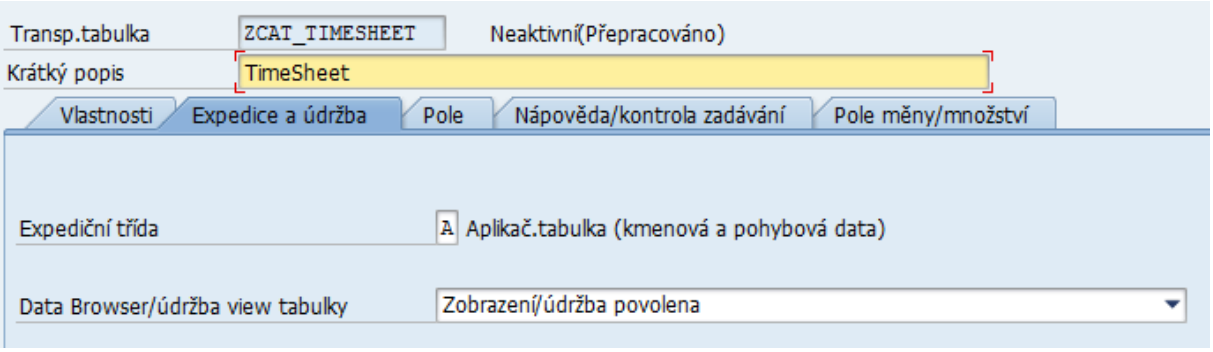
Kompletní návrh datového modelu je možné najít v příloze č. 2.

5.3 Realizace datového modelu

Pro realizaci řešení bylo nutné dle návrhu datového modelu tyto tabulky s jednotlivými prvky, případně i doménami vytvořit. Díky tzv. dopředné navigaci byla však tvorba těchto tabulek v podstatě otázkou času a znalostí systému. Důvodem je, že SAP již obsahuje spoustu předdefinovaných datových prvků a domén, které jsou již v samotném základu systému. Příkladem může být datový prvek pro klienta obsahující již předdefinovaný datový prvek MANDT s doménou CLNT o délce 3 znaků určený přímo pro klienta.

5.3.1 Tvorba tabulek, datových prvků a domén

Tabulky se vytváří v již zmíněném Object Navigator, který je popsán v kapitole 3. Při tvorbě datové tabulky se systém jako první dotáže na jméno tabulky a po jeho zadání je již možné vidět základní nastavení (viz obr. 7), které je nutné provést ještě před vytvořením jednotlivých prvků tabulky.



Obrázek 7: Tvorba tabulky ZCAT_TIMESHEET

Mezi toto základní nastavení patří:

Krátký popis – krátký popis, který je nutno uvést a zároveň při velkém množství tabulek slouží jako dobrá orientace.

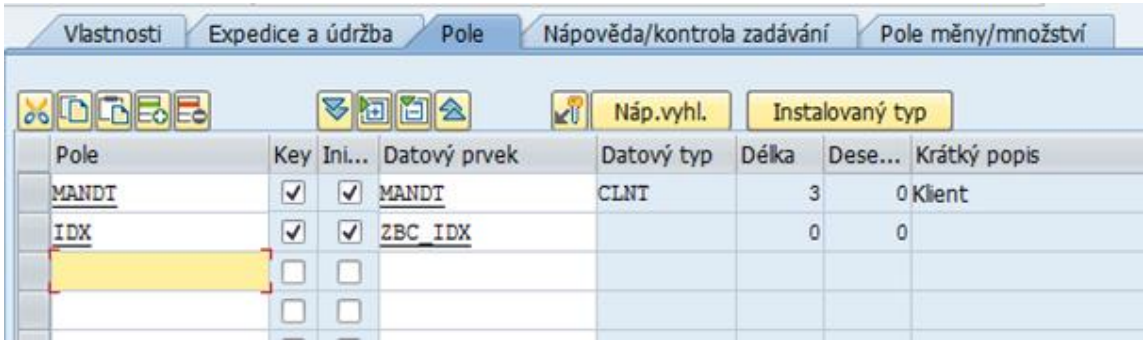
Expediční třída – *řídí přenos dat tabulky během instalací, aktualizací, kopírování klientů a přenosu do jiného systému SAP. Systém SAP a zákazníci mají různá práva k zápisu a to v závislosti na expediční třídě.* [1]

[1] Zdroj:KÜHNHAUSER, K. - H. *ABAP: Výukový kurz.* 1. vydání. 2009. ISBN 978-80-251-2117-7,str. 34

Data Browser/údržba view tabulky *– upravuje oprávnění pro prohlížení a údržbu dat se standardními nástroji pro údržbu dat v systému SAP.*[2]

V našem případě budou všechny tabulky obsahovat expediční třídu A – Aplikační tabulka (kmenová a pohybová data) – tedy tabulky, kde se data budou měnit jen zřídka a oprávnění zvolíme Zobrazení/údržba povolena - v opačném případě by nás systém nenechal vytvářet datové záznamy v objektovém slovníku.

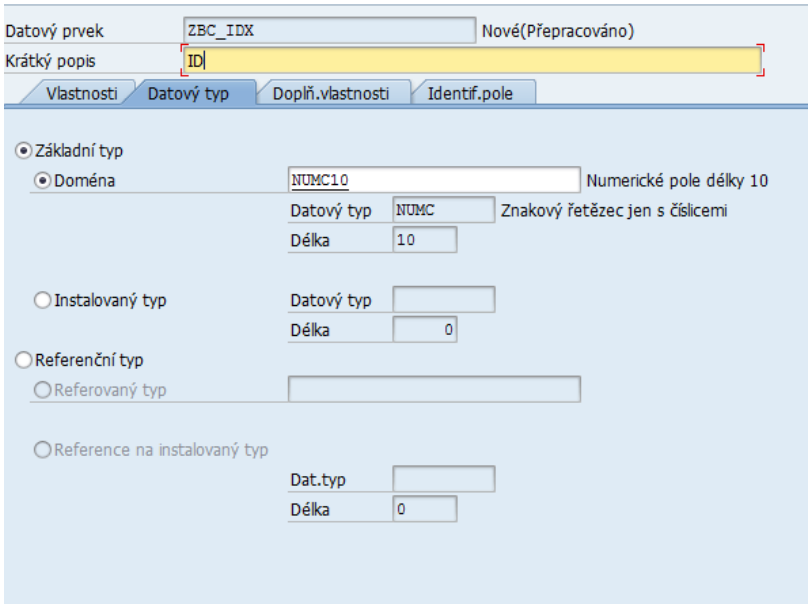
Poté je již možné přepnout na kartu Pole, kde se vyplňují jednotlivé prvky tabulky (viz obr. 8). Prvním záznamem každé naší tabulky je *Klient – MANDT*, který již SAP zná a po vyplnění prvku můžeme vidět v pravé části jeho datový typ, délku atd. Jelikož chceme, aby byl klíčovým prvkem a obsahoval inicializační hodnotu oba tyto checkboxy zaškrtneme. Dalším je *ID – IDX*, které je také klíčovým prvkem a také bude obsahovat inicializační hodnotu, ale zde si již musíme specifikovat datový prvek dle našich požadavků a při jeho zápisu se v pravé části zatím tedy neobjeví žádné informace o datovém typu.



Obrázek 8: Tvorba záznamů tabulky

[2] Zdroj:KÜHNHAUSER, K. - H. *ABAP: Výukový kurz.* 1. vydání. 2009. ISBN 978-80-251-2117-7,str. 34

V této chvíli bylo nutné datový prvek nejprve vytvořit a specifikovat a právě zde přichází na řadu právě dopředná navigace díky které stačí poklikat myší na zvolený datový prvek ZBC_IDX a tak okamžitě přejít k vytvoření tohoto prvku (viz obr. 6). Jelikož požadavkem je ID o velikosti 10 a pouze numerické hodnoty, zvolil jsem doménu NUMC10, která přesně odpovídá požadavkům pro ID a není tedy nutné další doménu zakládat.



Obrázek 9: Tvorba datového prvku

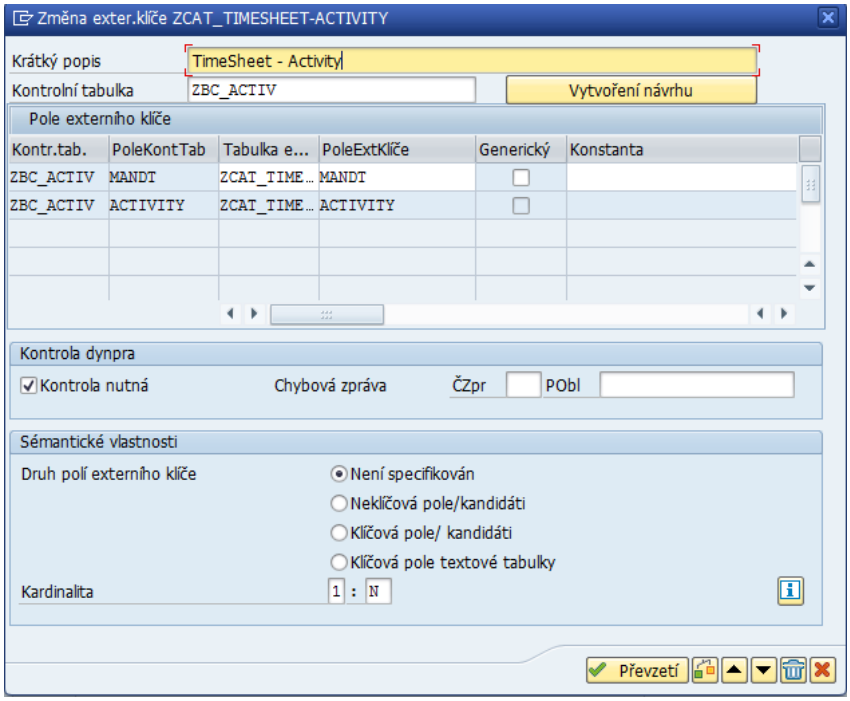
Dále se už musí vyplnit pouze Identifikátory pole, což jsou řetězce znaků různých délek, které se zobrazí v hlavičce sloupce (v závislosti na šířce tohoto sloupce).

5.3.2 Ukládání a aktivace

Ať už u tabulek nebo vytvořených datových prvků je nutné je vždy uložit a zde určit tzv „paket“, který závisí na tom zda chceme objekt převádět do jiného systému SAP či zda nám stačí uložen lokálně. Uložení však nestačí a bez následné aktivace je sice prvek uložen v paketu, ale systém o něm stále neví (v případě datového prvku bude pravá strana i po uložení stále prázdná), abychom daný objekt mohli v systému používat je tedy nutné ho také aktivovat.

5.3.3 Napojení tabulek – externí klíče

Mezi jednotlivými tabulkami bylo nutné vytvořit návaznost, která je docílena využitím externích klíčů. Tyto klíče propojují tabulky na základě uživatelsky vybraných prvků, které musí sdílet stejný datový prvek v opačném případě bude systém hlásit chybu. Při vytváření externích klíčů se také nastavuje kardinalita vztahu mezi danými tabulkami. Na obrázku 10 je možné vidět tvorbu externího klíče pro tabulku aktivit, která je současně kontrolní tabulkou, tedy tabulkou, ze které se data přenášejí.



Obrázek 10: Tvorba externího klíče

5.3.4 Nastavení aplikace

Jak už bylo napsáno výše, nastavení a kompletní customizing je realizován v samostatné transakci firemního frameworku. V této části je tedy provedeno nastavení vzhledu a chování aplikace. Prvním krokem bylo vytvoření klienta, pomocí kterého se celý customizing spravuje či ladí. Po vytvoření klienta bylo již možné určit tabulky, které bude aplikace obsahovat. V našem případě tedy všechny tabulky datového modelu. U každé tabulky je nezbytné vyplnit její obslužnou třídu a další atributy. Poté je již možné customizovat katalog polí, tlačítka nástrojové lišty a pole výběrové obrazovky. Tyto možnosti customizingu mají následující význam:

Katalog polí – Nastavení vzhledu a chování jednotlivých sloupců v ALV tabulce (mezi tato nastavení patří šířka sloupce, vzhled nadpisu nebo zarovnání textu)

Tlačítka nástrojové lišty – možnost nastavení standardních a zákaznických tlačítek (např. export dat do souboru)

Pole výběrové obrazovky – nastavení omezujících kriterií pro výpis dat tabulky (jaké položky chceme zobrazit)

Dále bylo nutné nastavit vhled menu a napojení tabulek na toto menu. Každá položka v menu obsahuje klíč uzlu, případně její nadřazený uzel, ikonu dle našeho výběru, číslo uzlu (pořadí ve kterém se zobrazí) a název tabulky, která se má otevřít při poklikání (viz obr. 11).

Menu

Klíč uzlu	Nadřazený uzel	Text	Ikona	Číslo uzlu	Název tabulky
UZEL1		TIMESHEET	@BH@	1	ZCAT_TIMESHEET
UZEL2		OBJEDNÁVKY	@03@	2	ZBC_PO
UZEL3		ZAKÁZKY	@9Z@	3	ZBC_ORDERHEADER
UZEL4		ČÍSELNÍKY		4	
UZEL5	UZEL4	MÍSTA	@AM@	1	ZBC_PLACES
UZEL6	UZEL4	AKTIVITY	@9Y@	2	ZBC_ACTIV
UZEL7	UZEL4	STATUS	@B0@	3	ZBC_STATUS
UZEL8	UZEL3	POLOŽKY	@AQ@	1	ZBC_ORDERITEM

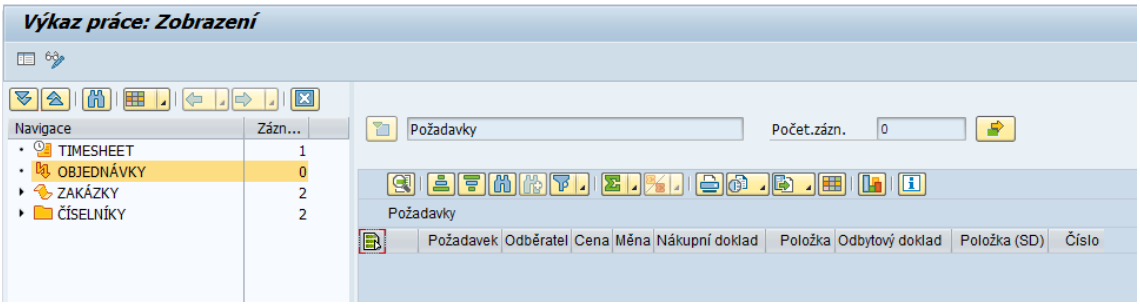
Obrázek 11: Tvorba menu pro aplikaci

V neposlední řadě se určují vazby mezi tabulkami, v našem případě je to právě tabulka hlavičky zakázky a tabulka zakázek, která bude její referenční tabulkou. Díky této vazbě je docíleno toho, že po výběru určité zakázky je možné vidět jednotlivé položky, které daná zakázka obsahuje a není tedy nutné projíždět všechny záznamy. Navíc zvolená zakázka z důvodu přehlednosti zůstane zobrazena v hlavičce aplikace.

5.3.5 Spouštění aplikace

Spouštění aplikace je obdobně jako většina aplikací v SAP systému spouštěna pomocí vlastní transakce, kterou bylo nutné vytvořit a opět nadefinovat její hodnoty.

Pro tvorbu transakce bylo jako první nutné zvolit kód transakce, přes kterou bude spouštěna a její popis. Jelikož chceme naši aplikaci spouštět na firemním frameworku bylo poté potřeba vytvořit návaznost mezi naší transakcí pro TimeSheet (ZBC_TS) s transakcí frameworku (ZIS). Ovšem tato návaznost nestačí, protože transakce ZIS může obsahovat více klientů s vlastním nastavením a customizingem proto po vyplnění spouštěné transakce bylo potřeba vyplnit tedy i klienta (BCT), který v této chvíli již obsahuje nadefinovaný vzhled a chování aplikace pro TimeSheet. Po uložení a aktivaci je již možné zadat kód transakce pro TimeSheet a je možné vidět celé rozhraní aplikace pro evidenci firemních dat (viz obr. 12).

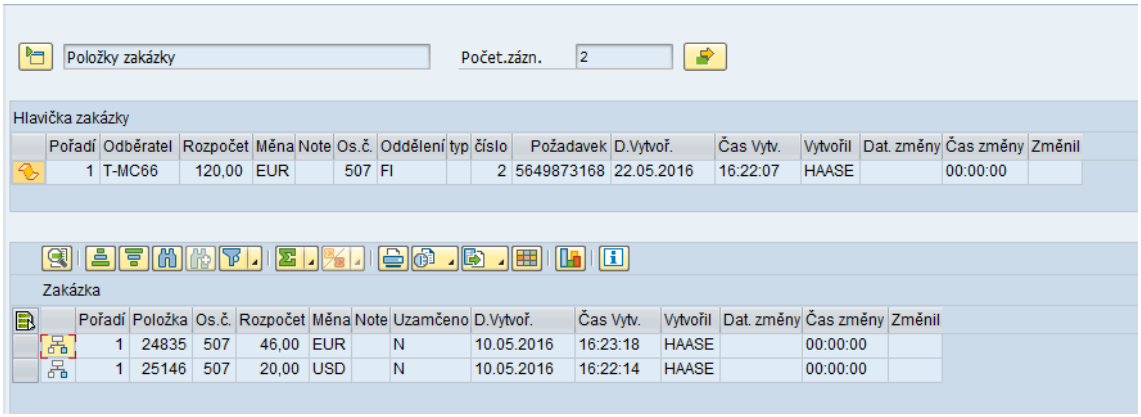


Obrázek 12: Výkaz práce - zobrazení

Na obrázku 12 je na levé straně možné vidět vytvořené menu – tabulky (timesheet, objednávky, zakázky, číselníky), které v této chvíli ovšem nedisponuje žádnými daty. Proto po výběru některé z položek se na pravé straně zobrazí pouze hlavička vybrané položky, nástrojová lišta s tlačítky a názvy jednotlivých sloupců, které vybraná tabulka obsahuje.

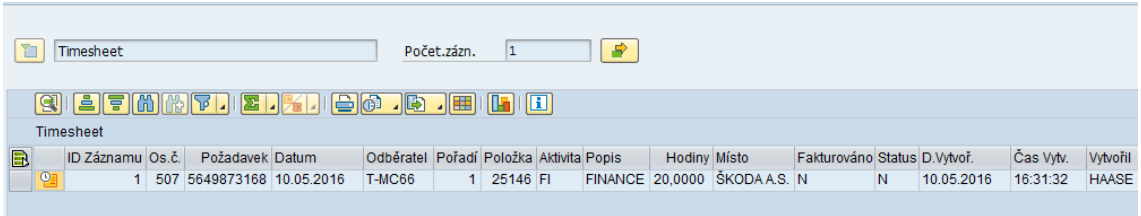
5.3.6 Testování aplikace

V aplikaci jsem vytvořil vlastní fiktivní data pro otestování celé aplikace a na obrázku 13 je možné vidět návaznost mezi tabulkou hlavičky zakázky a tabulkou položek zakázky, kde je návaznost dána polem „Pořadí“ (pořadové číslo zakázky). Horní tabulka je ALV tabulkou pro hlavičku zakázky, kde je možné vidět jednotlivé informace o dané zakázce. V dolní tabulce jsou již jednotlivé položky této zakázky a jejich doplňující informace.



Obrázek 13: Testování aplikace - Zakázky

Další, neméně důležitou částí pro testování byla tabulka TimeSheet, kde se na základě odpracovaných zakázek doplňují informace o tom, kdo zakázku zpracoval, stráveném čase na této zakázce atd. Po vyplnění těchto dat je již možné data poslat ke schválení zaměstnavateli, který již s nimi naloží dle svého nejlepšího uvážení. Na obrázku 14 je možné vidět nově vytvořený výkaz (status – N) pro naši první zakázku označenou číslem položky 25146.



Obrázek 14: Testování aplikace – TimeSheet

Po celkovém otestování je možné konstatovat, že vše funguje jak má a aplikace je připravena na naplnění firemními daty.

6 Závěr

Cílem této bakalářské práce bylo nastudovat podnikový informační systém od německé společnosti SAP. Prozkoumat a naučit se programovací jazyk ABAP, využívaný pro vývoj aplikací na platformě SAP a vytvořit aplikaci pro evidenci firemních dat v předem připraveném frameworku.

V první části práce jsem provedl rešerši na téma SAP jako podnikový informační systém (ERP), kde je možné najít informace o společnosti, historii, produktech a vývojovém prostředí vyvinutém touto společností. Dále je zde možné najít základní příkazy programovacího jazyku ABAP a to i s ukázkami funkčních kódů, které mi sloužily k jeho zvládnutí.

V další části jsem představil framework vyvinutý firmou T-MC66, s.r.o., který byl postaven na základě vlastních specifikací této firmy, a který je využíván jako prostředí pro návrh a tvorbu aplikací s návazností na standardní SAP objekty s možností variabilního nastavení podle potřeby plynoucí ze zadání aplikací.

V poslední části jsem detailně popsal návrh a realizaci vlastní aplikace pro evidenci firemních dat. Jde hlavně o vykazování odpracované činnosti pro firmu T-MC66, s.r.o. s návaznosti na zmíněný framework. Aplikaci jsem testoval zkušebními firemními daty tak, aby byla potvrzena její funkčnost.

Bakalářská práce mi dala příležitost naučit se pracovat v aplikačním prostředí SAPu, které je v České republice velmi rozšířené, ale na fakultě mechatroniky se v bakalářských oborech zatím nevyučuje. Znalost SAPu a souvisejících dovedností je praxí velmi žádaná a proto věřím, že jsem takto dostal příležitost lépe se uplatnit.

Použité zdroje a literatura

[1] *Plánování podnikových zdrojů* [online]. [vid. 25.4.2016].

URL:<https://cs.wikipedia.org/wiki/Pl%C3%A1nov%C3%A1n%C3%AD_podnikov%C3%BDch_zdroj%C5%AF>

[2] MAASE, A., GADATSCH, A., FRICK, D., SCHOENEN, M.

SAP R/3: Kompletní průvodce. 1. vydání. 2007. ISBN 80-251-1750-2.

[3] *SAP (společnost)* [online]. [vid. 19.8.2014].

URL: <https://cs.wikipedia.org/wiki/SAP_%28spole%C4%8Dnost%29>

[4] George W. Anderson: *Naučte se SAP za 24 hodin*, Computer Press, a.s., 2012

[5] *SAP: Příběh německých programátorů, kteří dobyli Wall Street* [online].

[vid. 10. září 2010]

URL: <http://www.itbiz.cz/sap-pribeh-nemeckych-programatoru-kteri-dobyli-wall-street>

[6] *SAP Česká republika* [online].

URL: < http://go.sap.com/cz/about.html#sapseglobal>

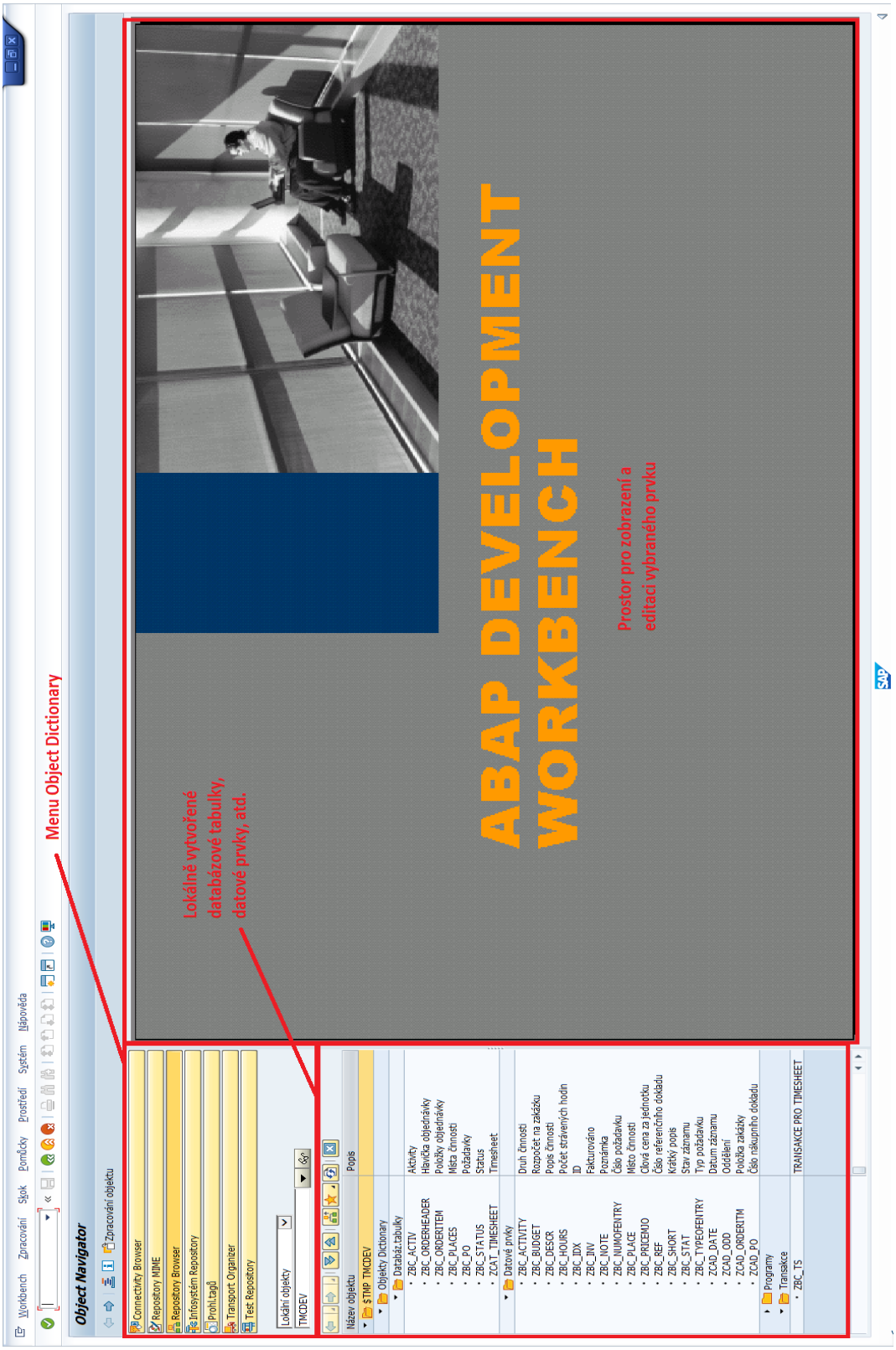
[7] KÜHNHAUSER, K. - H. *ABAP: Výukový kurz*. 1. vydání. 2009. ISBN 978-80-251-2117-7

[8] *SAP NetViewer* [online].

URL: < http://www.rankenen.cz/produkty/sap-netweaver/>

Seznam příloh

Příloha 1: Object navigator - popis



Příloha 2: Datový model