VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

PLÁNOVAČ VÝROBY TESTBEDU INDUSTRY 4.0
INDUSTRY 4.0 PLANNER

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Martin Kubásek
AUTHOR

VEDOUCÍ PRÁCE — Ing. František Burian, Ph.D.
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Automatizační a měřicí technika**

Ústav automatizace a měřicí techniky

Student: Martin Kubásek ***ID:*** 203270
Ročník: 3 ***Akademický rok:*** 2019/20

NÁZEV TÉMATU:

Plánovač výroby testbedu Industry 4.0

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je vytvořit adaptivní lokální plánovač výroby, který by naplánoval sekvenci výroby dle zadaných výrobních postupů a limitů pro malý testbed Industry 4.0. Výstupem plánovače by měla být spojová síť jednotlivých výrobních postupů vedoucí k hotovému výrobku, požadavků na materiál a přípravky, časový rozvrh výroby na jednotlivých strojích. Plánovač nemusí pracovat s kalendářem strojů, měl by ale být schopen reagovat na okamžité odstávky/servis.

1. Proveďte rešerši v názvosloví MES/ERP, zařazení plánovače do procesu.
2. Vytvořte jednoduchý plánovač splňující zadání
3. Vytvořte REST spojení s MES databází testbedu Industry 4.0 pro získání aktuálních úloh.
4. Spočítejte a uložte do souboru JSON výsledek Vašeho plánovače.

DOPORUČENÁ LITERATURA:

MEYER, Heiko, Franz FUCHS a Klaus THIEL. Manufacturing execution systems: optimal design, planning, and deployment. New York: McGraw-Hill, c2009. ISBN 978-0071623834.

Termín zadání: 3.2.2020 ***Termín odevzdání:*** 8.6.2020

Vedoucí práce: Ing. František Burian, Ph.D.

doc. Ing. Václav Jirsík, CSc.
předseda rady studijního programu

Abstrakt

Tato bakalářská práce se zabývá tvorbou plánovače výroby pro testbed Industry 4.0. Tvorbě předchází vysvětlení pojmů souvisejících s plánováním výroby – MES, ERP a Industry 4.0. Poté je nastíněna problematika plánování pro tento testbed a vysvětleno možné řešení tohoto problému, které bylo dodáno v rámci zadání. Následně je popsána struktura a algoritmy plánovače, který vznikl v rámci této práce a který toto řešení používá. Plánovač je realizován jako aplikace v jazyce C#, která na základě dodaných vstupů z MES vyrobí plán operací na strojích. Vstupy z MES jsou načítány přes REST spojení, výstup je realizován jako soubor JSON. Na konci práce jsou ukázány výsledky plánovače jako report v tabulce xlsx. Plánovač je schopný vytvořit výrobní plán pro testbed Industry 4.0.

Klíčová slova

Industry 4.0, plánování, rozvrhování, MES, ERP, REST

Abstract

This thesis deals with the design of manufacturing planner for Industry 4.0 testbed. Before design the terms Industry 4.0, MES, ERP are explained. Then the problems concerning this testbed are explained and a possible solution is shown. This solution was supplied as part of the assignment. Planner was made around this solution and its structure and algorithms were shown. Planner is written as a C# application, which generates manufacturing plan from MES input data. This input data is downloaded through REST connection, output is saved to a JSON file. Finally, the results of this planner are shown as a report in xlsx table. The planner can successfully generate a manufacturing plan for Industry 4.0 testbed.

Keywords

Industry 4.0, planning, scheduling, MES, ERP, REST

Citace

KUBÁSEK, Martin. Plánovač výroby testbedu Industry 4.0. Brno, 2020. Dostupné také z: https://www.vutbr.cz/studenti/zav-prace/detail/127094. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce František Burian.

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma „Plánovač výroby testbedu Industry 4.0“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu referencí na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení §11 a následujících autorského 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, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno 3.6.2020 .....
(podpis autora)

Poděkování

Na tomto místě bych chtěl poděkovat vedoucímu práce
Ing. Františku Burianovi, Ph.D za jeho odborné rady při konzultacích. Dále bych chtěl poděkovat své rodině a přítelkyni za podporu při studiu a psaní této práce.

Obsah

Seznam obrázků

Seznam tabulek

1 ÚVOD

V rámci této práce vznikne adaptivní plánovač výroby, který bude schopný komunikovat s existujícím MES, který je provozován u testbedu Industry 4.0 v areálu CEITEC v Brně. Před realizací bude provedena rešerše v názvosloví Industry 4.0, MES a ERP. Bude vysvětleno zařazení plánovače do výrobního procesu. Plánovač výroby bude realizován jako program napsaný v jazyce C# a bude dostatečně modulární pro budoucí rozšiřování. Bude komunikovat způsobem REST s nadřazeným MES. Z tohoto systému získá vstupní data, pomocí kterých spočítá výrobní plán a ten uloží jako soubor JSON. Výrobní plán pak bude moci MES použít pro nastavování výrobních prostředků na testbedu.

2 TEORETICKÝ ROZBOR

2.1 Průmysl 4.0

Jedno z často používaných označení pro období, které začalo přelomem tisíciletí a prudkým vzestupem internetových technologií, je „Age of disruption“. Tento pojem může být přeložen jako „Věk vyrušení“. Je to doba, ve které lze stále očekávat nějaké vyrušení a reálně nelze plánovat projekty na více než 5 let dopředu, protože každou chvíli může přijít nějaká zásadní změna. Ta je zpravidla způsobena nějakou novou technologií, nebo změnou trhu. Tyto rapidní změny jsou zapříčiněny velkým rozšířením digitálních technologií, které se navíc každým rokem dále zvyšuje. Digitální komunikační technologie umožňují lidem a firmám komunikovat napříč celou planetou v reálném čase a přenášet u toho velké množství dat. Stále výkonnější a úspornější výpočetní technologie umožňují využívat vysoký výpočetní výkon ve stále se zmenšujících počítačích. V takových podmínkách se pak rodí nové technologie a nové oblasti trhu rychleji než kdy dřív [1].

Jedním z důsledků této doby je vznik myšlenky Průmyslu 4.0. Tento pojem se zrodil v roce 2012 jako součást projektu, který měl zkoumat zvýšení konkurenceschopnosti Německa pomocí nových high-tech technologií [2]. Průmysl 4.0 se snaží o vyšší integraci pokročilých digitálních systémů v průmyslu. Následující podkapitoly čerpají hlavně z [3] a [4].

2.1.1 Důsledky

Ve výsledku by integrace technologií Průmyslu 4.0 měla umožnit několik věcí, které jsou v této „rychlejší době“ stále důležitější [1] [3]:

1) **Time to market** – čas na trh. Zkrácení času, který uběhne od začátku designu produktu po jeho uvedení na trh.

2) **Batch size one** – dávková výroba jednoho výrobku. Mělo by být možné vyrábět výrobky, které budou vysoce přizpůsobitelné zákazníkem. Tyto výrobky by pak neměly mít náklady vyšší než výrobky vyráběné po mnoha kusech. Ideálně by mělo být možné takto vyrábět i po jednom kusu.

3) **Efektivita ve využívání prostředků** – ve světě je stále větší nedostatek surovin. To má za následek zvyšování cen a zvyšování potřeby udržovat výrobu ekologickou.

2.1.2 Předpoklady

Aby bylo možné splnit výše uvedené body, je zároveň potřeba zajistit hlavně následující podmínky:

1) **Flexibilita** – je nutné umožnit podstatně větší flexibilitu ve vývoji produktů. To by umožnilo větší propojení mezi návrháři, výrobním procesem a ekonomy.

2) **Decentralizace** – z hlediska organizace je nutné zredukovat/zefektivnit organizační struktury, aby se zrychlil tok informací a zrychlilo rozhodování. Z hlediska výroby může být decentralizace aplikována tak, že jednotlivé části zásobovacího řetězce se budou moci navzájem nahrazovat napříč továrnou, skupinou továren nebo dokonce mezi továrnami různých firem. Zároveň by byla ve stejném smyslu sdílena provozní data. V takovém případě pak mluvíme o takzvaném Digitálním zásobovacím řetězci [5].

3) **Modularita** – nové systémy musí být mnohem více modulární, aby bylo možné rychleji reagovat na měnící se potřeby ve výrobě. Jejich jednotlivé části by mělo být možné nahrazovat a rozšiřovat bez zásadních nákladů. To může být dosaženo pouze intenzivní standardizací softwarových a hardwarových rozhraní.

4) **Sběr dat** – data z výrobních procesů by měla být sbírána za účelem monitorování, tvoření simulací a následných optimalizací.

K zajištění těchto podmínek byly následně definovány technologie, které by měly přispět ke splnění těchto předpokladů, a tím následně umožnit splnění cílů vytyčených v 2.2.

2.1.3 Technologie Průmyslu 4.0

Následující technologie (a technologické trendy) budou pravděpodobně využity v nových procesech, stavěných dle myšlenky Průmyslu 4.0. Následující seznam je seřazen podle důležitosti pojmů z hlediska citací dle rešerše [4]:

1) **Cyber-Physical Systems (CPS)** – systémy slučující funkce digitálního a fyzického světa. Příklad takového systému je například systém s digitálním dvojčetem. Pomocí intenzivního sběru dat je vytvářeno digitální dvojče reálného výrobního procesu, které ideálně popisuje jeho fyzickou předlohu. Může tak v digitálním prostoru ukazovat svá provozní data v reálném čase,

nebo třeba simulovat opotřebení stroje. To může usnadnit údržbu a pomůže odhalit problémy před tím, než nastanou.

2) **Internet of Things (IoT)** – Internet věcí, cílem je všechny snímače, akční členy, programovatelné automaty apod. připojit na společnou komunikační platformu, v tomto případě internet. To by mělo zaručit lepší spolupráci mezi těmito prvky. Zároveň lze dosáhnout snadnějšího sběru dat.

3) **Internet of Services** – Internet služeb. Cílem je poskytovat některé služby přes internet. Mezi tyto služby se mohou řadit například výrobní prostředky, které by mohly být takto pronajímány například za účelem krátkodobé zvýšení produkce.

4) **Smart Factory** – „chytré továrny" kompletně vybavené senzory a akčními členy, které budou řízeny autonomními systémy. Tyto autonomní systémy by měly být bez vnějšího zásahu schopny optimalizovat výrobní procesy a automaticky reagovat na problémy při výrobě. Takto by měly fungovat buď zcela autonomně, nebo tím pomáhat lidem ve výrobě. Za tímto účelem je potřeba intenzivně využívat data ze snímačů a akčních členů pro analýzu těchto procesů.

Jak tyto systémy spolupracují shrnuje výstižně Hermann, Pentek a Otto [6]: „V modulárně strukturovaných Smart Factories monitorují CPS fyzické procesy, vytvářejí jejich virtuální kopie a dělají decentralizované rozhodnutí. Přes Internet věcí (IoT) spolu CPS komunikují a spolupracují zároveň mezi sebou a mezi lidmi. Přes Internet služeb (IoS) jsou pak nabízeny a využívány služby uvnitř organizace a mezi organizacemi."

2.1.4 Plánování výroby v Průmyslu 4.0

Obecně je plánování výroby poslední část plánování před samotným nastavením strojů a zahájením výroby. Zpravidla jde o alokaci výrobních prostředků v čase na základě velkého množství vstupních informací, kde samotné operace a výrobní prostředky musí být důkladně popsány. Ve výsledku je plánování složitá kombinatorická úloha, kde jsou práce přidělovány strojům a každá práce se skládá z několika operací. Každá operace trvá nějaký čas a ten se může lišit dle použitého stroje. Každá operace má nějaké závislosti ve formě jiných operací. Také například některé stroje nemohou střídat jednoduše (v krátkém čase) vykonávané operace. Plánování výroby zpravidla cílí na to, aby celkový čas plánu byl co nejnižší [7] [8].

Plánovač výroby může těžit z prostředků Průmyslu 4.0, hlavně díky velkému množství dat z výrobních prostředků, které navíc může dostávat téměř v reálném čase. Díky tomu se otevírá cesta tzv. dynamickému plánování. Tento druh plánování počítá s neustálým narušování plánu během výrobního procesu, které může být způsobeno například poruchou stroje, nemocí pracovníka, nedostupností materiálu atd. Každé takové vyrušení vyvolá přepočítání výrobního plánu, který se musí přizpůsobit těmto novým okolnostem. Existuje několik strategií, jak přepočítávat výrobní plán. Jedna z nich je reaktivní plánování, které nevyrábí žádný fixní plán a plánuje pouze „krok dopředu“ na základě momentální stavu. Každá nová událost ve výrobním procesu vyvolá přepočet. Další strategie je například prediktivní - reaktivní plánování, která vyrobí kompletní plán výroby a upravuje jeho části pouze v případě vyrušení [9] [8] .

Důležitá vlastnost tohoto druhu plánování je, že počítá s rozpracovanou výrobou na výrobních prostředcích. Plánovač musí nový plán konstruovat tak, aby nedocházelo ke zbytečným operacím (zbytečný přesun výrobku, zbytečné zastavení probíhající operace). Vzhledem k tomu, že k přepočítávání bude docházet neustále, je vhodné ho vykonávat bez čekání na zásah člověka. Je zřejmé, že tento druh plánování velmi spoléhá na přesné informace o momentálním stavu výrobků, materiálů a výrobních prostředků. K tomu napomůže větší vertikální integrace systémů v Průmyslu 4.0. Časté a složité přepočítávání bude vyžadovat vysoký výpočetní výkon [8].

2.2 Standardy a modely MES/ERP

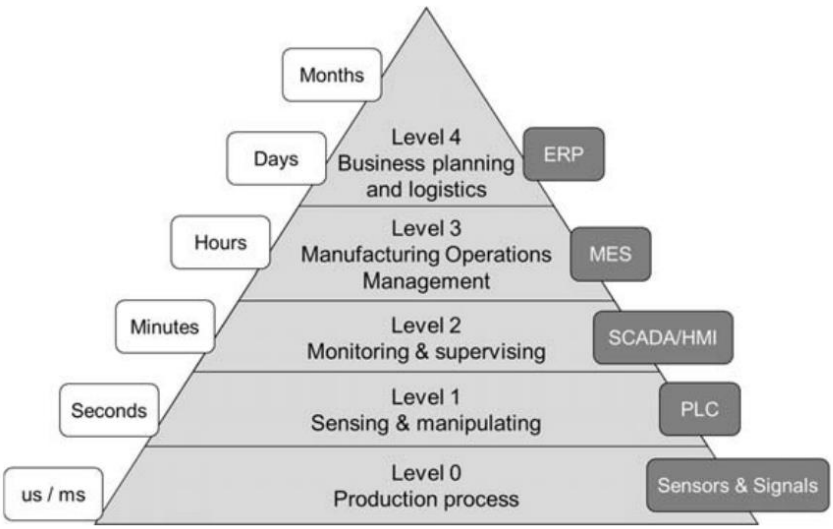
Před popisem MES a ERP systémů je vhodné alespoň stručně popsat, jaké standardy je popisují. Reálně ale nelze předpokládat, že všechny ERP a MES systémy jsou si mezi sebou rovny a že ctí tyto standardy. Každý systém bude vždy reflektovat subjektivní představu výrobce o tom, jak by takový systém měl vypadat. To je však docela přirozené, protože se mohou lišit i požadavky zákazníka [10]. Bude zde zmíněn standard ISA 95 a modely MESA.

2.2.1 ISA 95

Jedna z částí tohoto standardu popisuje úrovně jednotlivých systémů v průmyslu, jaké mají funkce a jak by měly mezi sebou komunikovat. Tyto systémy rozděluje podle druhu zpracovávaných informací do úrovní 0 až 4 [11]. Jednou z myšlenek (vertikální integrace) Průmyslu 4.0 je úrovně tohoto standardu více propojit a integrovat do sebe. Nakonec by tak vznikl jeden velký systém, který bude integrovat více úrovní do sebe. I tak je ale nutné samotné funkce nějakým standardem popsat [12].

2.2.1.1 Úrovně ISA 95

Úrovně tohoto standardu popisují funkce a názvosloví jednotlivých systémů ve výrobním procesu. Toto rozdělení také plyne z časových intervalů, se kterými tyto systémy pracují [13].



Obrázek 1 - úrovně ISA 95 [13]

Pro každou úroveň je zmíněn jen obecně nejznámější příklad. Ve skutečnosti může na každé úrovni existovat více druhů systémů. ERP a MES budou vysvětleny v dalších kapitolách.

Standard popisuje úroveň 0 až 4 [11]:

1) **Úroveň 4** – ERP
2) **Úroveň 3** – MES
3) **Úroveň 2** – SCADA, PLC
   Systémy přímo napojené na výrobní prostředek, spravující jeho sledování a nastavování. SCADA například může získávat data z PLC a nastavovat do PLC parametry. PLC jsou programovatelné automaty, které sbírají data ze snímačů procesu a nastavují svoje výstupy tak, aby ovlivnily chod procesu. Zajišťují automatické řízení.

4) **Úroveň 1** – Snímače, akční členy
   Snímače získávají data z výrobního procesu. Akční členy tyto procesy nějakým způsobem ovlivňují.

5) **Úroveň 0** – Výrobní proces

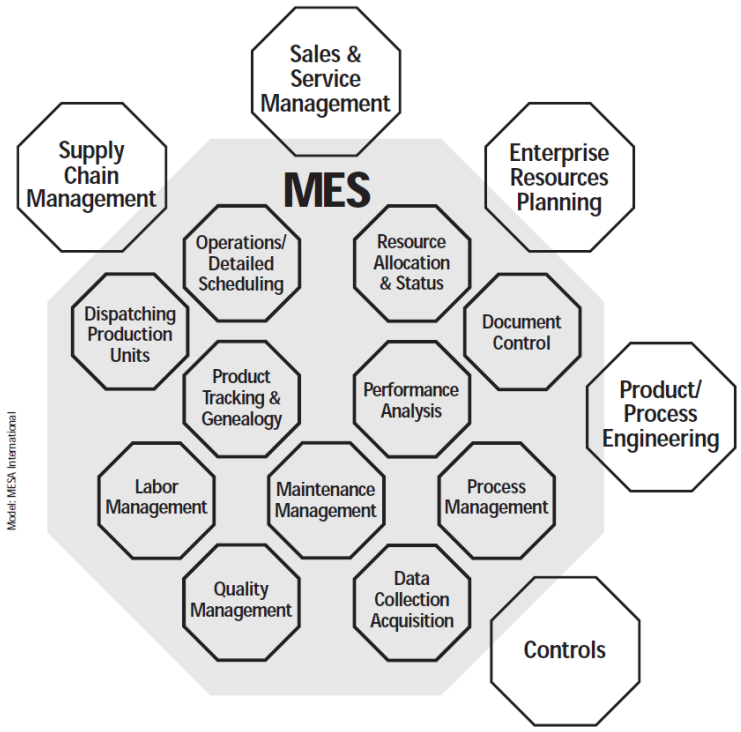
2.2.2 Modely MESA

Organizace MESA (Manufacturing Enterprise Solutions Association) International je nezisková organizace složena z několika výrobců MES systémů s cílem vylepšovat výrobní operace pomocí správné implementace informačních technologií a praktik managementu. Jedním z jejich cílů je tvořit modely pro informační systémy v podnicích a snažit se popsat vazby mezi nimi [14] [15].

2.2.2.1 MESA-11

V roce 1997 byl vytvořen model MESA-11, který poskytuje základní přehled o funkcích MES systémů. Tento model popisuje 11 základních funkcí, které by měl MES zajišťovat a poskytuje také obecný náhled na základní funkce MES [16]:

1) Detailní plánování operací
2) Rozdělování výrobních prostředků
3) Řízení lidských zdrojů
4) Řízení kvality
5) Řízení údržby
6) Řízení laboratoře
7) Řízení procesů
8) Správa dokumentace
9) Sběr dat
10) Sledování výrobků a jejich rodokmen
11) Statistika výkonnosti



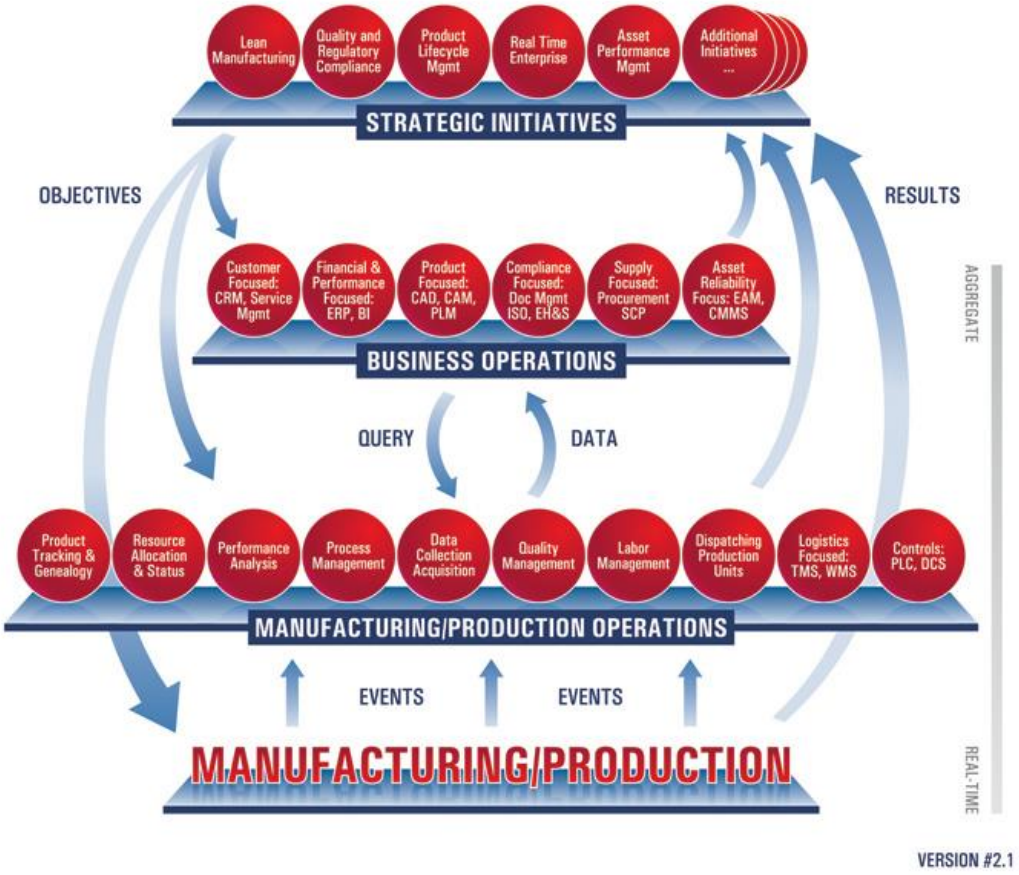
Obrázek 2- model MESA-11 [16]

2.2.2.2 C-MES

V roce 2004 vznikl nový model C-MES, který se snaží vylepšit propojení mezi systémy uvnitř podniku a propojení se systémy obchodních partnerů [15]. Dokument MESA [17] shrnuje tyto systémy: „C-MES systémy kombinují funkcionalitu původních MES systémů, tedy řídit a vylepšovat operace uvnitř továrny. Přidávají k tomu možnost lépe komunikovat s ostatními systémy a lidmi napříč zásobovacím řetězcem. I když byla některá data dříve sdílena pomocí běžných komunikačních prostředků, internet a technologie založené na webu umožňují výrazný skok v přesnosti a rychlosti všech komunikací.“

2.2.2.3 Stávající model MES

Nejnovější model MES z roku 2008 popisuje úrovně od výroby až po obchodní rozhodování. Ukazuje pohyb informací a vztahy mezi těmito úrovněmi [15]. Bylo by příliš rozsáhlé tento model shrnout v textu, obecný přehled poskytne následující obrázek.



Obrázek 3 - vztahy a pohyby informací dle nejnovějšího modelu MES [15]

2.2.2.4 Zařazení plánovače do výrobního procesu

Plánovač výroby musí být integrován jako součást úzce spolupracující s MES. Zjišťuje, v jakém stavu se nachází výrobní linka a plánuje, jaké budou další vykonávané operace. Z pohledu MESA-11 modelu potřebuje, aby mu MES zajišťoval funkce 2 - rozdělování výrobních prostředků, 7 - řízení procesů, 9 - sběr dat, 10 - sledování výrobků. Funkce 9 a 10 vyžaduje, aby bylo možné plánovat na základě reálného stavu výrobních linek. Funkce 2 a 7 plánovač od MES potřebuje, aby byl plán přidělen strojům/lidem. Se systémem MES si tak bude potřebovat vyměňovat velké množství informací.

2.3 MES

Obecně je MES (Manufacturing execution system) informační systém s cílem zvýšit efektivitu výroby, plánování prostředků a kontroly kvality. Spojuje do jednoho celku více menších systémů, které jsou obvykle nasazeny napříč výrobními procesy. Z těchto procesů sbírá v reálném čase data, která pak budou popisovat stav celé výroby. Monitoruje průběžně stavy strojů, pracovišť, skladů, materiálů ve výrobě. Z těchto kousků dat vytváří smysluplná sjednocená data, která mohou být použita k další analýze nebo k plánování. Kromě monitorování však zajišťuje i samotné nastavování výrobních prostředků. Umožňuje tak ucelený náhled na celý výrobní proces, ze kterého by mělo být například zřejmé, jestli jsou stroje plně funkční, zda se stíhá výrobní plán a vše je vyráběno a dokumentováno dle direktiv společnosti [16]. Obecně lze říct, že MES pracuje s daty týkajících se výroby, které se mění v řádech sekund až minut [13]. Z pohledu standardu ISA 95 jde o systém úrovně 3.

MES podle modelu MESA-11 obecně komunikuje se systémy ISA 95 úrovně 2 (SCADA) a 4 (ERP). Těmto systémům poskytuje svoje data a zároveň z nich data sbírá. Vyšší 4. úrovni může poskytovat data, jako například spotřebu materiálů, časy výrobních cyklů, statistika výkonnosti, evidence pracovníků na pracovišti, umístění materiálů. Nižší 2. úrovni pak může například poskytovat výrobní recepty, nebo plány výroby [16].

Novější MES systémy už by měly kopírovat model na obrázku 3. Zde by měl pak zařizovat všechny funkce na úrovni „Manufacturing/production operations“. V tomto případě by měl být MES více propojen s řídícími systémy a pokrývat tak ISA 95 úrovně 3 až 1, tedy být blíže propojen se samotnými výrobními prostředky. Toto větší propojení umožní větší flexibilitu a rychlost při řešení problémů ve výrobě. Také umožní nové metody plánování a řízení výroby, protože jsou data získaná z 2. úrovně přímo porovnávána s daty, která byla původně v jiném systému (z 3. úrovně). Naopak budou podle dat ze 3. úrovně ovlivňovány systémy, které byly původně odděleny [8].

2.4 ERP

Enterprise Resource Planning (ERP) systémy jsou informační systémy, které jsou zaměřeny na správu informací a prostředků ve firmách. V případě ERP (oproti MES) jsou zpracovávána data týkající se více samotného podnikání. Většinou jde o data, která se mění méně často, řádově hodiny až roky [13]. Integrované ERP systémy mohou být používány napříč celou firmou pro zlepšení efektivity všech procesů, které jsou ve firmě provozovány. Jeden ze základních principů, díky jimž je použití ERP systémů ve firmách výhodné, je funkce shromažďování informací na jediném centralizovaném místě. Díky tomu se snadněji přenášejí informace mezi pracovníky a je zaručeno, že všechna oddělení pracují vždy s nejaktuálnějšími daty [18].
Tento systém používá například [18]:

- Oddělení marketingu a prodeje pro správu objednávek, evidenci požadavků na podporu, analýzu dat o prodejích produktu
- Oddělení správy zásobovacího řetězce pro nákupy materiálů do výroby, evidence přijímaných a expedovaných materiálů/výrobků, správa logistiky, plánování výroby z dlouhodobého hlediska
- Oddělení účetnictví a financí pro správu faktur, vytváření a plánování rozpočtů
- Oddělení lidských zdrojů pro správu zaměstnanců (např. najímání, správa výplat), distribuci firemních materiálů

Z pohledu ISA 95 je ERP systém na nejvyšší úrovni 4. Z pohledu stávajícího modelu MESA (na obrázku 3) zařizuje funkce z kategorie „Bussiness operations“.

2.5 REST

Pojem Representational state transfer (REST) definuje architekturu pro vytváření webových služeb (Web services). Popisuje, jak by měly být webové služby postaveny, jakým způsobem jsou mezi nimi přenášena data a jakým způsobem jsou tato data zpřístupněna. Pokud je služba postavena dle REST architektury, nemělo by záležet v jakém jazyce byla napsána, ani jak se interně s daty zachází [19].

Tento styl vytvořil Roy Fielding v roce 2000, kdy zároveň vytvářel specifikaci pro HTTP 1.1. Pomocí REST se tak snažil popsat nejzákladnější principy, vlastnosti a omezení technologií využívající HTTP [20].

REST je definován několika hlavními vlastnostmi:

2.5.1 Rozdělení na server a klient

Tato architektura definuje situaci, kdy existují dva druhy zařízení – server a klient. Server je zařízení, které zpravidla běží neustále a čeká na požadavky od klienta. Když přijde požadavek od klienta, je nějakým způsobem na serveru zpracován a klientovi je odeslána odpověď. Tímto způsobem je pak možné, aby existoval jeden server obsluhující více klientů. Rozdělení se pak týká i samotných funkcí – například server se může zabývat pouze zpracováním dat a klient zobrazováním dat uživateli. Pokud se způsob komunikace mezi těmito zařízeními nezmění, mohou se nezávisle na sobě rozvíjet a měnit [21].

2.5.2 Bezestavovost

Požadavky, které jsou posílány klientem, musí obsahovat všechny potřebné informace k úspěšnému zpracování požadavku na serveru. To znamená, že server by neměl potřebovat si udržovat informaci o předchozích požadavcích. Veškerý kontext se tak drží pouze na straně klienta. Tento způsob vylepšuje spolehlivost – při neúspěšném přenosu se pouze zopakuje požadavek. Umožňuje serveru ihned uvolnit prostředky po vyřízení požadavku. Na druhou stranu mohou být kvůli tomuto způsobu některá data posílána zbytečně [21].

Aby se zmenšil objem přenášených dat, byl zaveden systém cachování. V odpovědi serveru by mělo být definováno, jestli je možné data v odpovědi na nějakou dobu ukládat, případně na jak dlouho. Klient si pak tyto data lokálně ukládá a nemusí se na ně dotazovat, dokud nevyprší maximální doba cache. To zase může způsobit nechtěné chování, kdy klient používá data, která již nejsou aktuální [21].

2.5.3 Jednotné rozhraní

Aby mohly rozdílné webové služby mezi sebou komunikovat, měly by používat jednotné rozhraní pro komunikaci. To zahrnuje [19]:

2.5.3.1 Použití HTTP dotazů

Služby fungující REST způsobem by měly využívat (klient), nebo podporovat (server) minimálně tyto HTTP dotazy:

GET pro stažení dat ze serveru
POST pro vytvoření dat na serveru
PUT pro editaci dat na serveru
DELETE pro smazání dat na serveru

Funkcionalitu těchto dotazů by služby neměly žádným nestandardním způsobem rozšiřovat – například vytvářet data na serveru po přijetí GET dotazu [19].

2.5.3.2 V rámci dotazů správně využívat Content-type

HTTP dotazy obsahují takzvanou hlavičku, ve které je definován Content-type. Ten označuje, o jaký druh obsahu se v dotazu jedná. Content-type pak může být například: text/html, text/xml, application/json, image/jpeg a mnoho dalších.

Správná REST aplikace by neměla nahrávat/stahovat obsah v nějaké nestandardní formě – například jako obsah parametru dotazu. Pro přesun dat by měla vždy správně používat jeden z definovaných typů, což je u REST aplikací nejčastěji JSON, XML a HTML. V praxi by pak měla aplikace i pro ty nejjednodušší PUT/GET dotazy používat data zabalená např. v JSON [19].

2.5.3.3 Přistupovat k datům s použitím URI

Aby byl přístup k datům jasně definován a nejlépe i čitelný člověkem, měly by REST služby přistupovat k datům pomocí URI. Tyto adresy by pak měly tvořit složkovou strukturu, kde jsou data hierarchicky rozděleny od nejzákladnějšího k nejpodrobnějšímu identifikátoru [19]. Pokud bychom například načítali z nějakého serveru data podle času, může URI definovat cestu k datům jako:

http://www.rest-priklad.org/testbed/historie/2020/2/28/data

3 NÁVRH PLÁNOVAČE

V této kapitole bude popsán datový model, jaká data vstupují do plánovače a jaká jsou jeho výstupem. Dále bude nastíněna problematika plánovače a popsána jeho architektura.

3.1 Problematika

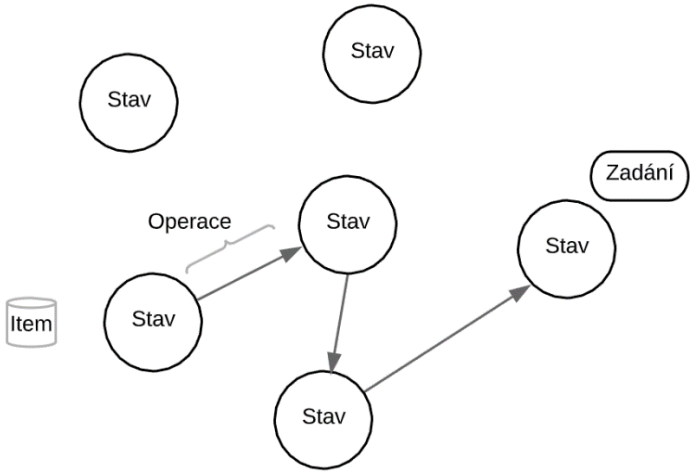
Plánovač bude s výrobními prostředky a s pohybem materiálů pracovat následujícím způsobem. Všechny materiály a výrobky jsou si rovnocenné a je na ně pohlíženo jako na předměty (Item). Z MES si plánovač načte, jaké předměty se na výrobní lince nacházejí a jaké jsou objednávky (Order). Každá objednávka je pak spárována s jedním až několika výrobky a je v ní definováno, jaký je cílový stav předmětu (opracování, poloha). Pokud existuje objednávka bez přiřazeného předmětu a zároveň existuje volný předmět, plánovač interně přiřadí tento předmět k objednávce.

Postupné opracování výrobku a změny jeho polohy jsou u každého předmětu zaznamenávány do atributu Labels a značí tak stav výrobku. Každý stav Labels připouští pouze několik dalších stavů. Možné změny stavu jsou pak definovány v operacích (Operation), které je možné na testbedu provést. Plánovač tak bude pro každý předmět hledat takové operace, které ho dostanou z jeho momentálního stavu Labels do stavu, který je definován v atributu Zadani objednávky.

Vzhledem k decentralizované struktuře výrobní linky může existovat více cest. Stroje (Machine) na výrobní lince jsou volně rozmístěny a předmět se může mezi nimi volně pohybovat. Navíc může existovat více strojů, které zvládnou provést stejnou operaci. Úkol plánovače bude tyto cesty najít a vytvořit z nich plán aktivit (Activity), podle kterých bude MES nastavovat stroje. Při plánování musí brát v potaz, že ne všechny operace budou dostupné, kvůli dostupnosti stroje. Stroj může být z nějakého důvodu mimo provoz (údržba, porucha).

Plánovač by měl být z MES periodicky spouštěn ve chvílích, kdy se nějakým způsobem změní situace na výrobní lince. To může být například dokončení operace, změna dostupnosti stroje, nový Item, nová objednávka. MES v tu chvíli očekává, že plánovač rozhodne, co se bude v dalším kroku kde a kdy spouštět. K tomu mu poskytne celkový přehled o stavu výrobní linky pomocí vstupních entit. Plánovač je dle REST bezestavový, takže si mezi voláním z MES nic neukládá a rozhoduje se pouze na základě těchto vstupních dat.

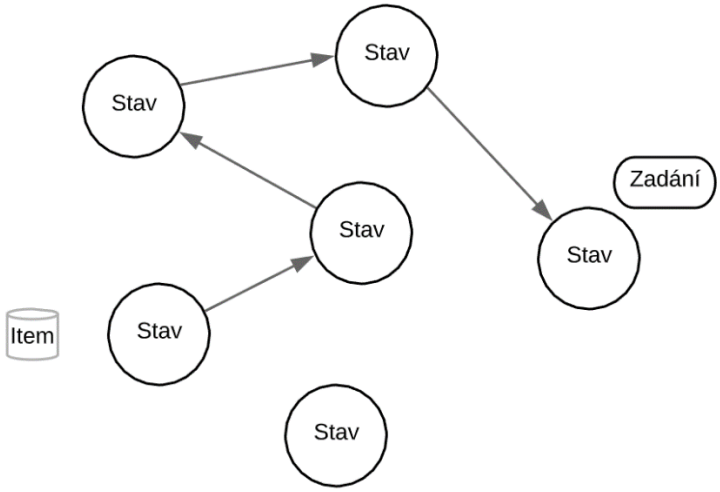
Pokud se pokusíme problematiku plánování zobrazit graficky, vypadalo by to takto:



Obrázek 4 - příklad nalezené cesty

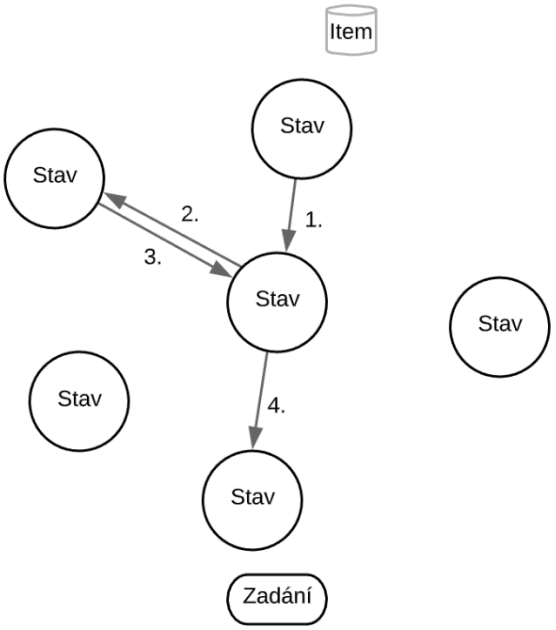
Zde se Item nacházel na nějakém místě na výrobní lince (zůstal v nějakém stavu) a hledá cestu do stavu, kterým splní zadání.

Může však také existovat jiná cesta:



Obrázek 5 – příklad nalezené delší cesty

Zde plánovač našel jinou cestu, pomocí které se dostal do cíle s použitím více operací. Tato cesta by pravděpodobně mohla být méně časově výhodná, ale plánovač ještě musí zvážit čas trvání operací a připravenost strojů.



Obrázek 6 - příklad nalezené cesty z jiného počátečního bodu

Zde je další příklad, kde je výchozí a cílová pozice Itemu jiná. Zároveň je pravděpodobně prostřední stav nějaký transport na mezisklad, protože se Item v něm bude nalézat více než jednou.

3.1.1 Labels

Entity Item obsahují pole Labels, které označuje, v jakém stavu opracování a na jakém místě se nachází daný Item. Každý Item musí dostat svoje Labels do stavu, který je popsán v poli Zadani objednávky, ke které je Item přiřazen. Stav Labels změní každá operace pomocí výrazu, který je v poli Expression. Příklad řetězce Labels:

"Labels": "freza,hrub"

V tomto řetězci jsou jednotlivé Labely odděleny čárkami a značí, jak byl již Item zpracován. Například Label „freza“ značí, že se Item nachází na fréze, „hrub“ značí že byl frézován nahrubo.

Každá operace může provést tři úpravy Labels:

1) Přidat Label
2) Vyměnit Label – smazat jeden až několik Labelů a přidat nové
3) Smazat Label

Pokud na výše zmíněný příklad aplikujeme operaci s následujícím výrazem:

"Expression": "freza,hrub/freza,hrub,*kapsahrube*"

Změní se jeho stav na:

"Labels": "freza,hrub,kapsahrube"

V tomto případě jsme provedli přidání Labelu. Pokud bychom dále aplikovali operaci s výrazem:

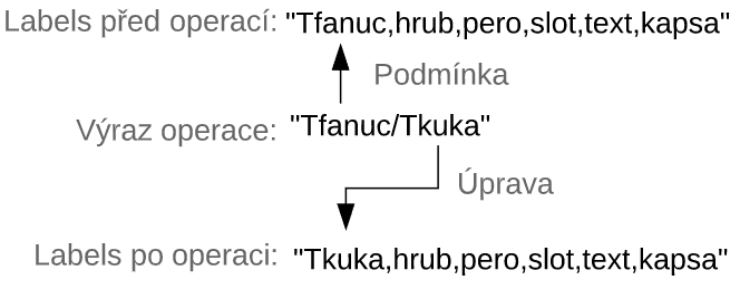
"Expression": "freza,hrub,kapsahrube/freza,hrub,kapsa"

Změní se jeho stav na:

"Labels": "freza,hrub,kapsa"

V tomto případě došlo k výměně Labelu.

Levá strana výrazu operací (před „/“) značí, které Labely musí být v Labels pro vykonání operace. Značí tedy podmínky vykonání operace. Pravá strana značí změnu v Labels po vykonání operace. Pravá strana je tedy nový stav podmínkových Labelů pro vykonání operace. Změna vyjádřená na pravé straně se týká vždy jen Labelů, které byly obsaženy v podmínce. Toto ukazuje následující obrázek:

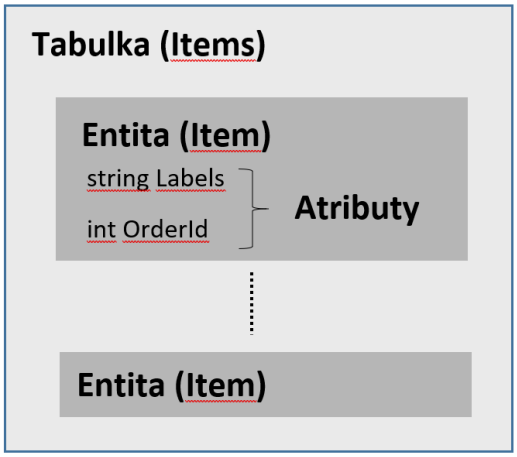


Obrázek 7 - ukázka zpracování výrazu

Na umístění Labelu nezáleží, pouze jestli je v řetězci obsažen. O jakou ze tří úprav jde, musí plánovač rozeznat. Dále je třeba počítat s tím, že některé Labely mohou mít parametry, například: „hrub=4.2“. Tyto parametry nemusí plánovač zohledňovat a z hlediska hledání cesty bude předpokládat, že jde pouze o „hrub“. U zápisu parametrů je použita u desetinného čísla tečka, protože čárkou jsou oddělovány jednotlivé Labely. Tento princip použití řetězců Labels byl také dodán v rámci zadání.

3.2 Vstup a výstup plánovače

V popisu vstupů a výstupu budou použity pojmy tabulka, entita a atribut. Jejich vztah popisuje následující obrázek.



Obrázek 8 - vztah tabulky entity a atributu

Tabulka obsahuje několik entit stejného druhu. Každá entita má pak svoje atributy (vlastnosti). Plánovač má 4 vstupní tabulky a 1 výstupní. Vstupem jsou tabulky Items, Machines, Operations a Orders. Výstupem je tabulka Activities. Zároveň existuje jedna přídavná entita PlannerCoreSettings, ve které jsou uloženy nastavení pro plánovač. Každá tabulka pak obsahuje entity se jménem podle tabulky v jednotném čísle (anglicky). Z programátorského hlediska lze v jazyce C# považovat entity za objekty (instance tříd).

Vstupy budou načítány z databáze nadřazeného MES pomocí REST spojení. Výstup plánovače bude tabulka Activities, která bude serializovaná ve formátu JSON a uložená do souboru. Podoba vstupních entit byla poskytnuta už v rámci zadání. Entita Activity vznikla v rámci této práce.

Dále jsou popsány jednotlivé entity a jejich vlastnosti. Všechny entity obsahují atribut číslo **Id**, což je primární klíč v rámci každé tabulky. Pro větší přehlednost už nebude zmiňována při popisu atributů entit.

3.2.1 Entita Order

Tento vstupní objekt popisuje objednávku na výrobu z MES. Z hlediska plánovače není důležitá vlastnost „Zdroj“.

Tabulka 1 - atributy entity Order

datový typ	jméno	popis
int	Stav	číslo označující stav objednávky, plánovač pracuje pouze s objednávkami, kde je Stav rovný 0
DateTime	Ukonceno	čas ukončení zpracování objednávky
string	Zadani	popisuje cílový stav Labels, do kterého se musí Item dostat, tedy popisuje stav, ve kterém je Item už hotový výrobek
DateTime	Zadano	čas zadání objednávky do MES
string	Zdroj	informace pro MES

3.2.2 Entita Item

Tento vstupní objekt popisuje výrobky/materiály, které se nachází na výrobní lince. Z pohledu plánovače nezáleží, jestli jde pouze o neopracovaný materiál nebo už je to něco, co lze považovat za výrobek. Proto jsou tyto objekty popsány obecně jako Items – předměty.

Tabulka 2 - atributy entity Item

datový typ	jméno	popis
string	Labels	řetězec popisující, co daný Item absolvoval, jak byl opracován a kde se nachází, toto bude popsáno podrobněji v následujících kapitolách
int	OrderId	číselný identifikátor propojující daný Item s objednávkou (Order), může existovat i více Itemů přiřazených k jedné objednávce

3.2.3 Entita Machine

Tento vstupní objekt popisuje stroje nebo obecně výrobní prostředky, které se nachází na výrobní lince. Pro plánovač není důležitý atribut „Name“.

Tabulka 3 - atributy entity Machine

datový typ	jméno	popis
int	Job	číslo označující Id operace, kterou zrovna stroj vykonává
DateTime	JobEndsAt	čas ukončení stávající operace
string	Name	jméno stroje
int	State	číslo označující stav stroje

Pro potřeby plánování byl u entity Machine zaveden lokální plán naplánovaných operací v čase. Ten slouží třídě BasicTimePlanner.

3.2.4 Entita Operation

Tento vstupní objekt popisuje jednotlivé operace, které lze na Itemu vykonat. Každá operace je vykonávaná určitým strojem a po aplikaci operace na Item se změní jeho pole Labels. To značí, že se změnila úroveň jeho opracování, nebo se změnila jeho poloha. Pro plánovač není důležitý atribut „Code“.

Tabulka 4 - atributy entity Operation

datový typ	jméno	popis
string	Code	informace pro MES
int	Duration	délka trvání operace v minutách
string	Expression	řetězec, který udává, jakým způsobem a za jakých podmínek se změní Labels u Itemu po aplikaci této operace
int	MachineId	číslo udávající Id stroje, který je schopný tuto operaci vykonat
string	Name	název operace

3.2.5 Entita Activity

Tento výstupní objekt zadává do MES informaci, jaká operace se bude aplikovat na jaký Item a v jaký to bude čas. MES bude podle této tabulky pak přímo nastavovat stroje.

Tabulka 5 - atributy entity Activity

datový typ	jméno	popis
int	ItemId	číslo udávající Id Itemu, na který se bude aplikovat operace
int	OperationId	číslo udávající Id operace, která se bude aplikovat
int	DependentOn	číslo udávající Id aktivity, která musí být vykonána před touto aktivitou
DateTime	StartTime	čas plánovaného spuštění operace na stroji
DateTime	EndTime	čas plánovaného dokončení operace na stroji

3.2.6 Entita PlannerCoreSettings

Slouží k nastavování jádra plánovače. Obsahuje převážně nastavení pro REST spojení.

Tabulka 6 - atributy entity PlannerCoreSettings

datový typ	jméno	popis
float	BranchTimeThreshold	Pouze pro BasicPathPlanner, hraniční čas u hledání cesty, vyjádřen jako procenta překročení nejnižšího času cesty
string	ServerUrl	Základ adresy REST spojení (serveru MES)
int	ServerPort	Port REST spojení
string	ItemsPath	Konkrétní cesta k entitě Items
string	MachinesPath	Konkrétní cesta k entitě Machines
string	OrdersPath	Konkrétní cesta k entitě Orders
string	OperationsPath	Konkrétní cesta k entitě Operations
string	ActivitiesPath	Konkrétní cesta k entitě Activities
bool	IsWriteEnabled	Povolení k zápisu do MES

3.3 Realizace

Plánovač výroby byl napsán v jazyce C# 7.3 s cílovou architekturou .NET Framework 4.5.2. Byly vytvořeny tři projekty: But.Planovac, But.Planovac.Core a But.Planovac.Core.Tests.

Při tvorbě plánovače byl kladen velký důraz na modularitu řešení. Nejdůležitější třídy byly napsány jako abstraktní třídy. Ty obsahují funkcionalitu, u které lze očekávat, že bude vyžadována vždy. Tyto třídy zároveň obsahují abstraktní metody, které v tomto případě plní funkci rozhraní, tedy nutí implementační třídu rozšířit tyto metody. Samotné algoritmy jsou obsaženy až v konkrétní implementaci těchto tříd. Využití abstraktních tříd umožní snadnější vývoj nových algoritmů a zvýší celkovou čitelnost kódu. Celkem takto vznikly tři abstraktní třídy: *LoggerBase*, *PathPlannerBase*, *TimePlannerBase*. Všechny jsou obsaženy v jádru plánovače.

3.3.1 But.Planovac

Je obalový program, který volá metody v *But.Planovac.Core* a slouží jako program pro testování jádra plánovače. Je realizován jako konzolová aplikace, ve které lze měnit parametry plánovače a ovládat jeho vstup a výstup. Po spuštění se program zeptá, zda chce uživatel načíst data z MES, nebo jestli mají být použita testovací data.

Pokud uživatel vybere testovací data, proběhne načtení tabulek Orders, Items, Machines a Operations ze souboru. Program očekává, že ve stejné složce, kde byl

spuštěn, je také složka DbDump. Ta obsahuje soubory „<jméno tabulky>.json“. Tyto soubory obsahují vyexportované tabulky z MES, serializované do JSON formátu (textu). Tabulky v JSON formátu je následně třeba deserializovat do náležitých entit. K tomu složí třída *JsonHandler*, která je obsažená v *But.Planovac.Core* projektu. Tyto entity jsou pak zvláštní metodou podstrčeny *But.Planovac.Core.* Zároveň je zakázáno připojení k MES, takže neproběhne uložení výstupu do souboru JSON[1].

Při výběru načítání z MES je *But.Planovac.Core* spouštěn běžným způsobem a proběhne navázání REST spojení s MES. Z nadřazeného systému si plánovač stáhne všechny tabulky, které jsou také ve formátu JSON. Stejně jako u načítání ze souboru, proběhne v *But.Planovac.Core* deserializace.

Po výběru způsobu načítání dat začne proces plánování. Jádro plánovače načte ze složky, kde bylo spuštěno, soubor settings.json. V tomto souboru je JSON serializovaná entita PlannerCoreSettings, která obsahuje nastavení pro jádro. V průběhu plánování tiskne jádro přes logovací třídu do okna konzole postupně aktuální stav. Po skončení procesu se program zeptá, jestli uživatel chce vygenerovat report do .xlsx souboru, nebo chce spustit proces znovu. V této chvíli, nebo při výběru dat lze měnit klávesovou zkratkou ALT a +/– časový práh pro hledání cest. Toto se týká pouze implementace s BasicPathPlanner.

V report v .xlsx souboru (tabulka Microsoft Excel) je zobrazen výsledek plánování pro každý Item. Jde o přímou interpretaci výstupu plánovače – tabulky Activity. Podrobnější popis je v kapitole výsledků.

3.3.2 But.Planovac.Core

Obsahuje veškerou logiku kolem plánování, jedná se o jádro plánovače. Je realizován jako knihovna tříd (.dll soubor). Příklad jeho ovládání je v metodě *StartPlanner* v obalovém projektu. Jeho hlavní řídící třída je *PlannerController*, kterou je potřeba vytvořit a předat jí třídu dědící z *LoggerBase*. Tato logovací třída slouží k okamžitému informování o průběhu plánování. Záleží na konkrétní implementaci, kde se tyto informace zobrazují. V obalovém projektu se vypisují na konzoli prostřednictvím třídy *ConsoleLogger*.

Dále je potřeba zavolat metodu *Execute*. Tato metoda vrací výčtový typ, z kterého je možné dedukovat, jestli proces plánování proběhl bez problémů (bez ohledu na samotný obsah plánování).

[1] Původní zadání práce počítalo i s přímým zápisem do MES. Nyní je toto chování simulováno generováním výsledků do JSON souboru.

Informuje nás o následujících výsledcích:

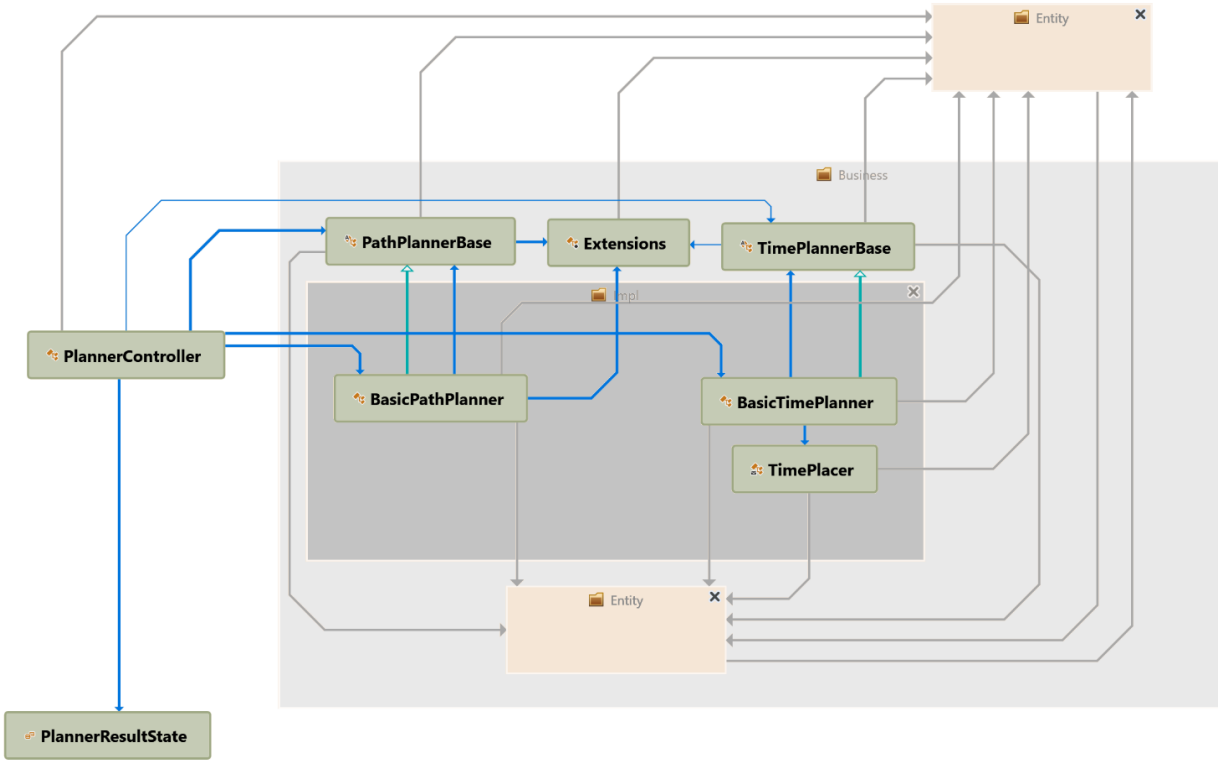
1) nebylo možné načíst vstupní data
2) úspěšně vytvořil nějaký plán
3) vytvořil plán, ale nepodařilo se ho uložit
4) nevytvořil žádný plán

Metoda *Execute* vykoná několik operací:

1) Načte nastavení.
2) Pokusí se načíst vstupní data. Pokud je to povoleno, získává data spojením s MES. Jinak se pokusí použít testovací data.
3) Vytvoří třídu dědící z *PathPlannerBase*
4) Zavolá metodu *AssignItemsToOrders*. Ta najde objednávky bez přiřazených Itemů a pokusí se jim přiřadit volný Item.
5) Zavolá metodu *GetAllPathPlans*. Ta se pokusí najít každému Itemu možné cesty po výrobní lince.
6) Vytvoří třídu dědící z *TimePlannerBase*, které předá nalezené cesty.
7) Zavolá v třídě z bodu 5 metodu *GetNextActivities.* Ta vybere každému Itemu nejlepší cestu po výrobní lince a vytvoří plán aktivit pro nadcházející výrobu. Vrací entity Activity.
8) Získané Activity serializuje do formátu JSON a uloží. Připraví se pro případný export do xlsx.

Plánování výroby se odehrává v bodech 5 a 7. Konkrétní implementace zmíněných tříd jsou *BasicPathPlanner* a *BasicTimePlanner*. Rozdělení na hledání cesty a na hledání času je další forma modularity. Při vývoji nového algoritmu pro hledání cesty po výrobní lince nemusí být potřeba kopírovat kód pro hledání vhodného času operací a naopak.

Následující obrázek popisuje architekturu tříd v jádru plánovače. Zelená šipka značí závislost třídy a modrá značí volání metod.



Obrázek 9 – závislosti týkající se plánování v projektu

3.3.2.1 Načítání nastavení

Před připojením k MES dojde k načtení nastavení jádra ze souboru. Jádro očekává, že tam kde bylo spuštěno, se bude nacházet soubor settings.json. Konkrétní funkce jednotlivých atributů je popsána v kapitole Vstup a výstup plánovače, z většiny jde o parametry REST spojení. Soubor s příponou „.json“ je prakticky textový soubor, který může obsahovat například následující nastavení:

```
{
  "planningSettings": {
    "BranchTimeThreshold": 1.25
  },
  "connectionSettings": {
    "ServerUrl": "ricaip.ceitec.vutbr.cz",
    "ServerPort": 2223,
    "ItemsPath": "/items",
    "MachinesPath": "/machines",
    "OrdersPath": "/orders",
    "OperationsPath": "/operations",
    "ActivitiesPath": null,
    "IsWriteEnabled": false
  }
}
```

Obrázek 10- příklad settings.json

Tento text jádro deserializuje pomocí třídy JsonHandler a vznikne tak jedna instance třídy PlannerCoreSettings, se kterou se dále pracuje.

3.3.2.2 RestHandler

Tato třída zajišťuje načítání dat z MES. Toto načítání je prakticky poslání GET dotazu na server MES, ve kterém je specifikována cesta k datům. Na tento dotaz pak server odpoví posláním textu, který reprezentuje JSON serializovanou entitu, kterou jsme si vyžádali. Princip byl také popsán v kapitole 2.5. Tato entita se poté deserializuje do objektu, s kterým dále můžeme v programu pracovat. Při neúspěchu je vrácen prázdný objekt. Neúspěch pak většinou značí nedostupnost serveru.

Celkově takhle stahujeme ze serveru 4 entity (Item, Machine, Order, Operation). Aby byl proces co nejrychlejší, je načítání paralelizováno do 4 vláken.

3.3.2.3 PathPlannerBase a BasicPathPlanner

Obrázek 11 - vztah PathPlannerBase

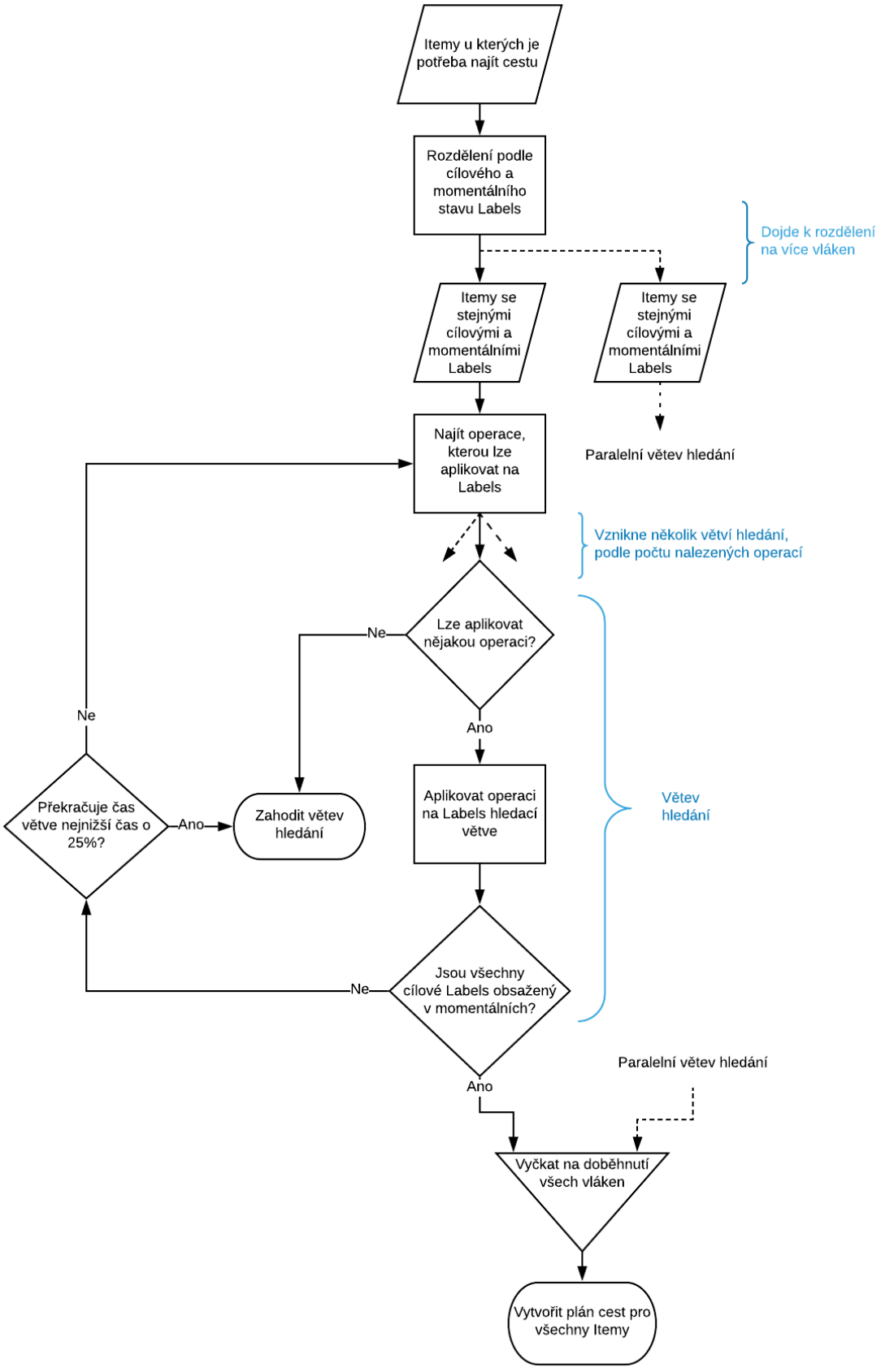
Tato třída se zabývá hledáním cesty po výrobní lince. Pomocí informací o stavu Itemů a o možných operacích vygeneruje plány možných cest. Základní třída obsahuje již dříve zmíněné *AssignItemsToOrders.* Dále obsahuje pomocné metody pro (před)zpracování dat – získání relevantních objednávek, předzpracování Labels. Nejdůležitější funkcionalita základní třídy je podpora paralelního zpracování a optimalizace na základě výchozího a cílového stavu. Dojde k seskupení všech zpracovávaných Itemů dle jejich stávajícího stavu a dle jejich cílového stavu. Protože by nalezené cesty pro tyto Itemy byly identické, budou hledány jen jednou v rámci skupin. Dále dojde k paralelnímu hledání cest. Pro každou skupinu probíhá hledání ve vlastním vlákně. Vznikne maximálně tolik vláken, kolik má hostující počítač logických procesorů.

Pro zvýšení přehlednosti kódu byly přesunuty často používané metody do třídy Extensions. Zde jsou tak na jednom místě důležité operace jako porovnávání řetězců Labels, oddělování parametrů od Labelů, aplikování operací na řetězec Labels.

Implementace ve formě *BasicPathPlanner* obsahuje jednoduchý algoritmus, ve kterém jsou pomocí rekurze aplikovány operace na původní Item. Po každém aplikování operace je porovnáváno, jestli jsme splnili zadání, tedy Labels obsahuje všechny cílové Labely. Pokud větev splnila všechna zadání, je přidána do seznamu

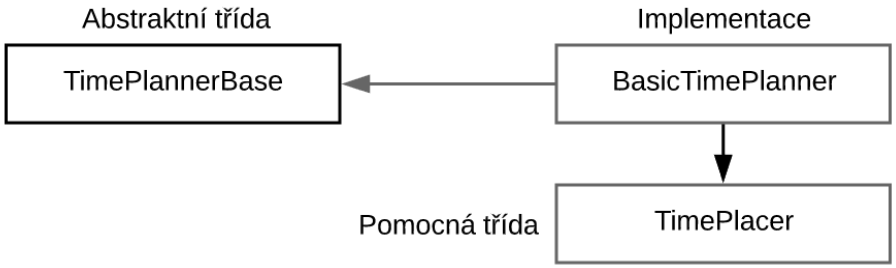
nalezených větví. Při každém vnoření je spočítána celková doba větve, což je součet trvání všech aplikovaných operací na této cestě. Pokud je její čas nejnižší, je zaznamenán. Protože z každého stavu Labels existuje více než jedna operace, kterou lze dále použít, rozvětvuje se toto hledání do velkého počtu větví. Každá větev vytvoří tolik dalších větví, kolik lze použít dalších operací. Aby bylo zamezeno zacyklení a byl zlepšen výkon hledání, je omezeno použití již použitých operací. Pokud je čas větve o 25 % vyšší než doposud nejnižší nalezený čas větve, je hledání na této větvi ukončeno. Metoda vrací plán cest pro každý Item, ve kterém jsou cesty seřazeny podle času vzestupně. Algoritmus hledání cesty pro každý Item je graficky znázorněný na následujícím vývojovém diagramu.

Celkově tedy provádíme hledání skupiny možných cest od začátku do cíle v grafu, ve kterém známe pouze sousední uzly. Kdybychom použili neupravený Dijkstrův algoritmus, získali bychom pouze jednu nejkratší cestu. Jak ukážu později, není tohle zcela praktické z hlediska plánování výroby, protože nejkratší cesta nemusí být proveditelná.



Obrázek 12 - algoritmus hledání cesty v BasicTimePlanner

3.3.2.4 TimePlannerBase a BasicTimePlanner

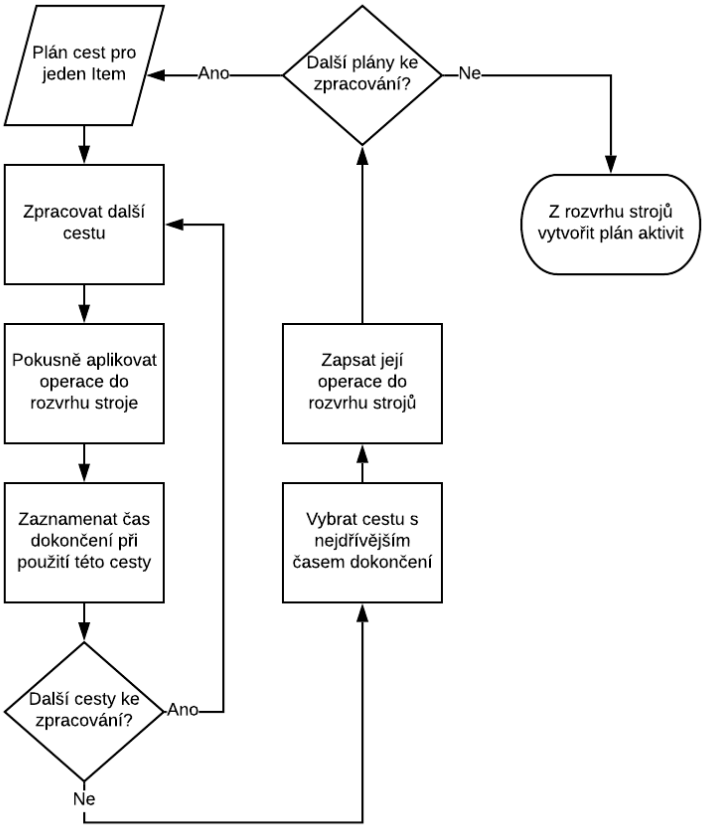


Obrázek 13 - vztah TimePlannerBase

Tato třída se zabývá hledáním časového plánu a generováním seznamu entit Activity. Pomocí informací o strojích a nalezených cest vytváří časový plán vykonávaných operací na strojích. Tento plán poté převede na seznam Activity, podle kterého již MES může nastavovat stroje. Základní třída obsahuje jedinou veřejnou metodu, kterou je potřeba volat pro získání seznamu Activity. Ta zajišťuje volání metod pro přípravu plánu operací na strojích, umístění potřebných operací na stroje a převedení na seznam Activity. Dále tato třída vynucuje implementaci potřebných neveřejných metod.

Implementace ve formě *BasicTimePlanner* zajišťuje přípravu plánu operací na strojích. Pokud na stroji již probíhá nějaká operace, přidá se do plánu operací jako první. Dále postupně zpracovává jednotlivé plány cest. Plán cesty obsahuje odkaz na Item a nalezené cesty. Algoritmus se pokouší nastavovat jednotlivé operace na stoje pro všechny nalezené cesty v rámci plánu cest. K tomu používá pomocnou třídu *TimePlacer*. Do plánů operací na strojích zatím nic neukládá, pouze si zaznamenává celkový čas dokončení. Až zpracuje všechny cesty v rámci plánu cest, vybere ten s nejdřívějším časem dokončení a aplikuje ho na stroje. Dojde k uložení do plánů operací strojů. Další zpracovávaný plán se tedy musí přizpůsobit dříve umístěnému plánu. Po rozmístění plánů na stroje se generuje seznam Activity.

To, že má BasicTimePlanner k dispozici více cest, je výhodné z hlediska optimalizace. Dle algoritmu na obrázku 13 si vyzkouší všechny tyto cesty dosadit do rozvrhů strojů a zjišťuje, s kterou cestou bude nejlepší čas dokončení. Protože na strojích už pravděpodobně bude něco naplánovaného, nebudou si cesty se stejným časem operací rovnocenné (viz tabulka 8), protože jedna z cest bude lépe využívat kapacitu strojů. Do určité míry může být užitečné zkoušet dosazovat cesty, které obsahují zdánlivě zbytečné operace. Jsme také do budoucna schopni se přizpůsobit kalendáři strojů, protože známe téměř všechny možné cesty a nejsme tak pevně vázáni na dostupnost jednoho stroje.



Obrázek 14- algoritmus tvoření Activit v BasicPathPlanner

3.3.3 But.Planovac.Core.Tests

Tato třída obsahuje jednotkové testy pro testování jádra plánovače. Jednotkové testy slouží k otestování jednotlivých funkcí programu. Vznikly testovací třídy pro *PlannerController*, *BasicPathPlanner* a *BasicTimePlanner*. I v tomto případě je zachována modularita pomocí abstraktních tříd, kdy pro plánování cesty a času existuje vždy základní a implementační testovací třída. Většina testování probíhá v základních třídách. V implementačních třídách je jen nezbytné minimum kódu.

V rámci testování je vždy kousku kódu poskytnut vstup, který by měl zaručit nějaký známý výsledek. Porovnává se, jestli výstup tohoto kusu kódu odpovídá očekávanému správnému výsledku. Pro testování plánování jsou zde umístěny textové soubory s JSON entitami, které tvoří vstup plánovače. Je zde testováno jestli:

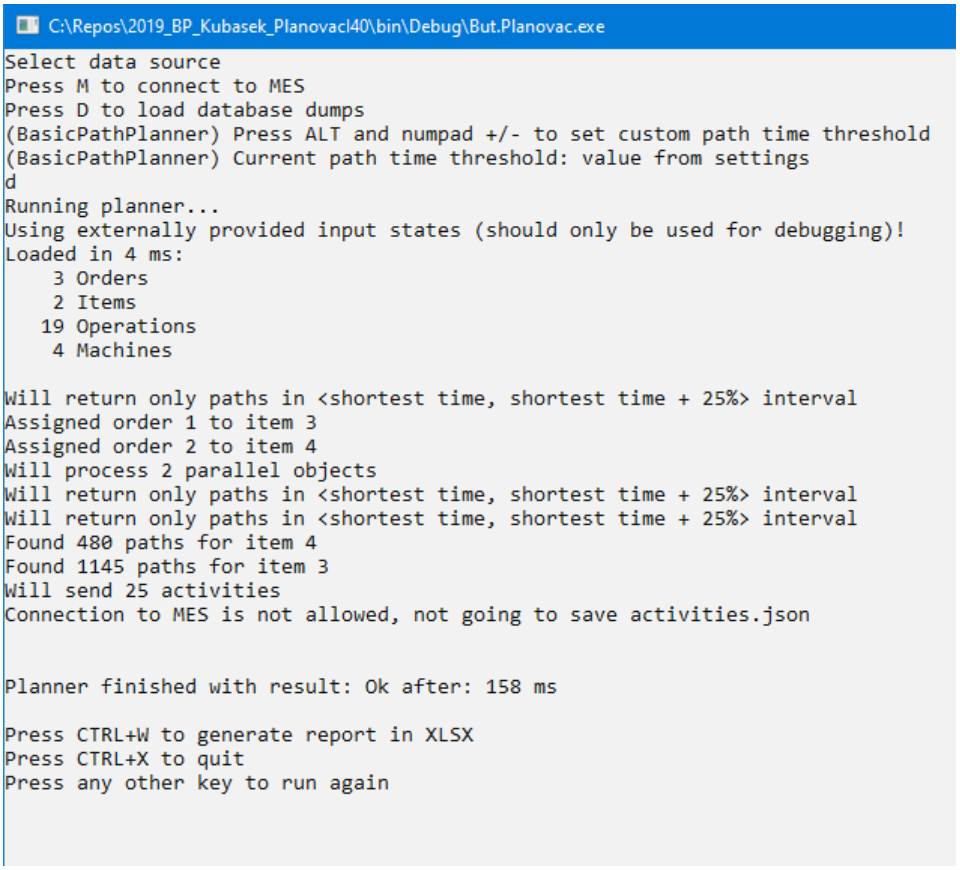
1) Konstruktory tříd reagují správně na špatné parametry.
2) Jsou správně načítána testovací data.
3) Plánovač najde plán výroby, když to má být možné.
4) Plánovač nenajde plán výroby, když to nemá být možné.

5) Dochází k správnému přiřazování Itemů k objednávkám.
6) Dochází ke správnému rozřazování do skupin v *BasicPathPlanner*.
7) Program se dokáže vypořádat se špatně zadanými daty a reaguje na tuto situaci dle očekávání.

Celkem za účelem testování vzniklo 45 jednotkových testů. Tyto testy umožní rychle ověřit, že nové algoritmy (implementace PathPlannerBase a TimePlannerBase) jsou napsány správně a chovají se dle očekávání.

3.4 Výsledky plánovače

Proběhlo spuštění plánovače pomocí obalového projektu – konzolové aplikace. Plánovač úspěšně načetl testovací data, přiřadil objednávku k Itemu, našel cesty i časový plán. Poté bylo možné udělat report, ve kterém byl zobrazen seznam Activity. Plánovači byly poskytnuty testovací data ve formě textových souborů obsahujících tabulky z MES, serializované do formátu JSON. Spuštění plánovače pomocí konzolové aplikace a její výstup vypadá takto:



Obrázek 15 - Spuštění plánovače jako konzolová aplikace

Na následujícím obrázku je JSON reprezentace tabulky Items se dvěma entitami Item, které si plánovač načetl:


```
[
  {
    "Id": 3,
    "Labels": "fanuc",
    "OrderId": -1
  },
  {
    "Id": 4,
    "Labels": "sklad",
    "OrderId": -1
  }
]
```


Obrázek 16 - JSON serializovaná tabulka Items

Itemy byly přiřazeny k těmto objednávkám:


```
{
  "Id": 2,
  "Stav": 0,
  "Ukonceno": null,
  "Zadani": "text=aaa,hrub,slot,pero,privesek",
  "Zadano": "2019-02-01T00:00:00Z",
  "Zdroj": null
},
{
  "Id": 1,
  "Stav": 0,
  "Ukonceno": null,
  "Zadani": "text=aaa,hrub,slot,pero,privesek",
  "Zadano": "2019-01-01T00:00:00Z",
  "Zdroj": null
}
```


Obrázek 17 - JSON serializované entity Order

3.4.1 Výstup do souboru JSON

Primární výstup plánovače je realizován jako seznam Activity, který je serializován do formátu JSON a uložen jako textový soubor. Text, který je uložen v tomto souboru, by mohl být v budoucnu nahrán pomocí REST do MES pomocí metody POST.

Část serializované tabulky Activity ve výstupním souboru pak vypadá takto:


```
{
    "DependentOn": 3,
    "EndTime": "2020-06-03T15:48:27Z",
    "Id": 7,
    "ItemId": 3,
    "OperationId": 4,
    "StartTime": "2020-06-03T15:38:27Z"
  },
  {
    "DependentOn": 2,
    "EndTime": "2020-06-03T14:18:27Z",
    "Id": 8,
    "ItemId": 4,
    "OperationId": 13,
    "StartTime": "2020-06-03T14:08:27Z"
  }
```


Obrázek 18 - JSON serializovaná entita Activity

3.4.2 Report v xlsx tabulce

Do jádra plánovače byla zabudována možnost prezentovat výsledky plánování jako report v tabulce xlsx. Soubor reportu obsahuje dva listy – Items a Paths. List Items interpretuje vytvořený seznam Activity, takže ukazuje data, která již může použít MES. List Paths poté ukazuje pro každý Item všechny cesty, které byly nalezeny (kromě těch, které už PathPlanner zahodil). Následující tabulka je zkopírována z listu Items. Ukazuje naplánované operace pro Item (seznam Activity) začínající s Labels „sklad“.

Tabulka 7 - plán aktivit pro Item s Id=4

Výraz operace	Start	Konec	Labels po vykonání operace	Popis operace
			sklad	Výchozí stav
sklad/fanuc	13:39	13:49	fanuc	Vyjmi ze skladu
fanuc/freza	13:49	13:54	freza	Vloz do frezy
freza/freza,hrub	13:54	14:04	freza,hrub	Hrubovani
freza,hrub/freza,hrub,pero	14:04	14:09	freza,hrub,pero	Frezovani pera
freza,hrub/freza,hrub,slot	14:09	14:14	freza,hrub,pero,slot	Frezovani slotu vizitka
freza,hrub/freza,hrub,text	14:14	14:24	freza,hrub,pero,slot,text	Frezovani textu
freza,hrub/ freza,hrub,kapsahrube	14:24	14:29	freza,hrub,pero,slot,text,kapsahrube	Frezovani otvoru freza 10
freza,hrub,kapsahrube/ freza,hrub,kapsa	14:29	14:34	freza,hrub,pero,slot,text,kapsa	Frezovani otvoru freza 2
freza/fanuc	14:34	14:44	fanuc,hrub,pero,slot,text,kapsa	Vyjmi z frezy
fanuc/Tfanuc	14:44	14:54	Tfanuc,hrub,pero,slot,text,kapsa	Vloz na transport
Tfanuc/Tkuka	14:54	15:04	Tkuka,hrub,pero,slot,text,kapsa	Transport Fanuc-Kuka
Tkuka,kapsa/Tkuka,privesek	15:04	15:07	Tkuka,hrub,pero,slot,text,privesek	Kompletuj klicenku
Tkuka/Tvydej	15:07	15:17	Tvydej,hrub,pero,slot,text,privesek	Transport Kuka-Vydej

Plánovač tak úspěšně naplánoval Activity pro dosažení zadání objednávky. Tato tabulka ukazuje ideální stav. Žádný stroj nebyl nijak vytížen a operace tak následují ihned po sobě.

Druhý Item se už musel přizpůsobit aktivitám prvního:

Tabulka 8 - plán aktivit pro Item s Id=3

Výraz operace	Start	Konec	Labels po vykonání operace	Popis operace
			fanuc (výchozí stav)	
fanuc/freza	13:54	13:59	freza	Vloz do frezy
freza/freza,hrub	14:34	14:44	freza,hrub	Hrubovani
freza,hrub/freza,hrub,pero	14:44	14:49	freza,hrub,pero	Frezovani pera
freza,hrub/freza,hrub,slot	14:49	14:54	freza,hrub,pero,slot	Frezovani slotu vizitka
freza,hrub/freza,hrub,text	14:54	15:04	freza,hrub,pero,slot,text	Frezovani textu
freza,hrub/ freza,hrub,kapsahrube	15:04	15:09	freza,hrub,pero,slot,text,kapsahrube	Frezovani otvoru freza 10
freza,hrub,kapsahrube/ freza,hrub,kapsa	15:09	15:14	freza,hrub,pero,slot,text,kapsa	Frezovani otvoru freza 2
freza/fanuc	15:14	15:24	fanuc,hrub,pero,slot,text,kapsa	Vyjmi z frezy
fanuc/Tfanuc	15:24	15:34	Tfanuc,hrub,pero,slot,text,kapsa	Vloz na transport
Tfanuc/Tkuka	15:34	15:44	Tkuka,hrub,pero,slot,text,kapsa	Transport Fanuc-Kuka
Tkuka,kapsa/Tkuka,privesek	15:44	15:47	Tkuka,hrub,pero,slot,text,privesek	Kompletuj klicenku
Tkuka/Tvydej	15:47	15:57	Tvydej,hrub,pero,slot,text,privesek	Transport Kuka-Vydej

Po prozkoumání výsledků zjistíme, že oba výrobky putovaly po výrobní lince od určité chvíle paralelně. Největší zpoždění vzniklo na fréze, na kterou musel druhý výrobek čekat 35 minut. Data, s kterými plánovač v tomto příkladu pracoval, byla vytvořena pouze jako příklad pro vyzkoušení funkcionality plánovače. To se týká i času všech operací.

Následující tabulky jsou zkopírovány z listu Paths. Je možné na nich ukázat, jak mohou vypadat rovnocenné cesty z hlediska času. V tomto případě začínal Item s Labelem „sklad“:

Tabulka 9 - ukázka rovnocenných cest

varianta	1	2	3
operace	1 Vyjmi ze skladu	1 Vyjmi ze skladu	1 Vyjmi ze skladu
	2 Vloz do frezy	2 Vloz do frezy	2 Vloz do frezy
	13 Hrubovani	13 Hrubovani	13 Hrubovani
	18 Frezovani pera	18 Frezovani pera	18 Frezovani pera
	17 Frezovani slotu vizitka	16 Frezovani textu	16 Frezovani textu
	14 Frezovani otvoru freza 10	17 Frezovani slotu vizitka	14 Frezovani otvoru freza 10
	15 Frezovani otvoru freza 2	14 Frezovani otvoru freza 10	17 Frezovani slotu vizitka
	16 Frezovani textu	15 Frezovani otvoru freza 2	15 Frezovani otvoru freza 2
	3 Vyjmi z frezy	3 Vyjmi z frezy	3 Vyjmi z frezy
	4 Vloz na transport	4 Vloz na transport	4 Vloz na transport
	8 Transport Fanuc-Kuka	8 Transport Fanuc-Kuka	8 Transport Fanuc-Kuka
	7 Kompletuj klicenku	7 Kompletuj klicenku	7 Kompletuj klicenku
	10 Transport Kuka-Vydej	10 Transport Kuka-Vydej	10 Transport Kuka-Vydej
Čas [min]	98	98	98

V této tabulce lze vidět cesty, které mají stejný čas operací (součet délky operací). Je vidět, že minimálně operace číslo 14, 16 a 17 na sobě nejsou nijak závislé a mohou proběhnout v jakémkoliv pořadí.

Na dalším příkladu můžeme ukázat, jak může vypadat neefektivní cesta:

Tabulka 10 - ukázka neefektivní cesty

operace	1 Vyjmi ze skladu
	2 Vloz do frezy
	13 Hrubovani
	14 Frezovani otvoru freza 10
	15 Frezovani otvoru freza 2
	16 Frezovani textu
	17 Frezovani slotu vizitka
	18 Frezovani pera
	3 Vyjmi z frezy
	4 Vloz na transport
	8 Transport Fanuc-Kuka
	9 Transport Kuka-Fanuc
	11 Transport Fanuc-Vydej
	20 Transport Vydej - Kuka
	7 Kompletuj klicenku
	10 Transport Kuka-Vydej
Čas [min]	123

Tato cesta obsahuje několik zbytečných operací na konci, kdy byl Item transportován zbytečně mezi všemi stroji. Nalezení této cesty ale nemusí být úplně bezcenné z hlediska plánování. Pokud bychom se museli vyhnout nějakému jinému výrobku na výrobní lince, tak by ji algoritmus pro hledání časového plánu mohl vybrat.

Kompletní podoba reportu v xlsx a všechny JSON serializované entity, které byly použity v zhodnocení výsledků plánování, jsou v příloze.

4 ZÁVĚR

Byla provedena literární rešerše ohledně pojmů MES, ERP a zařazení plánovače ve výrobním procesu. V rámci této rešerše byla také nastíněna myšlenka Průmyslu 4.0. Vznikl základní plánovač výroby napsaný v jazyce C#, který úspěšně dokáže najít cestu po testbedu Industry 4.0 za pomocí vstupních dat, která získá z nadřazeného MES. Tato data si dokáže stáhnout pomocí REST spojení, nebo je možné je načíst ze souboru. Na základě vstupních dat poté najde všechny vhodné cesty výrobků po testbedu a vytvoří časový plán aktivit. Tento plán aktivit lze považovat za výrobní plán, který definuje, co se bude kde a kdy provádět na testbedu. Plán aktivit je poté uložen do souboru JSON a je ho také možné lokálně vizualizovat jako xlsx tabulku. Z analýzy výstupních dat plánovače je pak zřejmé, že plán aktivit úspěšně tvoří spojovou síť výrobních postupů v čase, které vedou k hotovému výrobku. Plánovač je postaven takovým způsobem, že je schopen se adaptivně přizpůsobovat připravenosti strojů (odstávka/servis). Zároveň se dokáže přizpůsobit i aktuálním operacím, které jsou na strojích nastavené.

5 REFERENCE

[1] SCHWAB, Klaus. The Fourth Industrial Revolution, by Klaus Schwab | World Economic Forum. *World Economic Forum* [online]. 2017. Dostupné z: doi:978-1-5247-5886-8

[2] FEDERAL MINISTRY FOR ECONOMIC AFFAIRS AND ENERGY. Plattform Industrie 4.0. *Plattform Industrie 4.0*. 2019.

[3] LASI, Heiner, Peter FETTKE, Hans Georg KEMPER, Thomas FELD a Michael HOFFMANN. Industry 4.0. *Business and Information Systems Engineering* [online]. 2014, **6**(4), 239–242. ISSN 18670202. Dostupné z: doi:10.1007/s12599-014-0334-4

[4] HERMANN, Mario, Tobias PENTEK a Boris OTTO. Design Principles for Industrie 4.0 Scenarios: A Literature Review. *Technische Universitat Dortmund* [online]. 2015. Dostupné z: doi:10.1109/HICSS.2016.488

[5] MUSSOMELI, Adam, Doug GISH a Stephen LAAPER. *The rise of the digital supply network* [online]. 2016 [vid. 2019-11-09]. Dostupné z: https://www2.deloitte.com/us/en/insights/focus/industry-4-0/digital-transformation-in-supply-chain.html

[6] HERMANN, Mario, Tobias PENTEK a Boris OTTO. Design principles for industrie 4.0 scenarios. In: *Proceedings of the Annual Hawaii International Conference on System Sciences* [online]. 2016. ISBN 9780769556703. Dostupné z: doi:10.1109/HICSS.2016.488

[7] FRAMINAN, Jose M., Rainer LEISTEN a Rubén RUIZ GARCÍA. *Manufacturing Scheduling Systems* [online]. 2014. Dostupné z: doi:10.1007/978-1-4471-6272-8

[8] ROSSIT, Daniel Alejandro, Fernando TOHMÉ a Mariano FRUTOS. Industry 4.0: Smart Scheduling. *International Journal of Production Research* [online]. 2019, **57**(12), 3802–3813. ISSN 1366588X. Dostupné z: doi:10.1080/00207543.2018.1504248

[9] LI, Jun Qing, Quan Ke PAN a Kun MAO. A discrete teaching-learning-based optimisation algorithm for realistic flowshop rescheduling problems. *Engineering Applications of Artificial Intelligence* [online]. 2015. ISSN 09521976. Dostupné z: doi:10.1016/j.engappai.2014.09.015

[10] KLETTI, Jürgen. *Manufacturing execution systems - MES* [online]. 2007. ISBN 9783540497431. Dostupné z: doi:10.1007/978-3-540-49744-8

[11] SIEMENS. *ISA 95 Framework & Layers* [online]. [vid. 2019-11-12]. Dostupné z: https://www.plm.automation.siemens.com/global/en/our-story/glossary/isa-95-framework-and-layers/53244

[12] BACHELOR, Michael. *ISA-95 Is Necessary for Smart Manufacturing* [online]. 2017 [vid. 2019-11-12]. Dostupné z: https://www.automationworld.com/products/networks/blog/13317924/isa95-is-necessary-for-smart-manufacturing

[13] ÅKERMAN, Magnus. *Implementing Shop Floor IT for Industry 4 . 0 Implementing Shop Floor IT for Industry 4 . 0 Department of Industrial and Materials Science*. 2018. ISBN 9789175977522.

[14] MESA INTERNATIONAL. *About MESA* [online]. Dostupné

z: http://www.mesa.org/en/aboutus/aboutmesa.asp
[15] MESA INTERNATIONAL. *MESA Model* [online]. [vid. 2019-11-17]. Dostupné z: http://www.mesa.org/en/modelstrategicinitiatives/MESAModel.asp
[16] MESA INTERNATIONAL. MES Explained: A High Level Vision. *MESA Internacional*. 1997, (6), 23.
[17] FEMIA, Dave, Jonathan KALL, Julie FRASER, Charlie GIFFORD a Karsten Newbury and Khris KAMMER. *MESA White Paper #08* [online]. 2014. Dostupné z: https://services.mesa.org/ResourceLibrary/ShowResource/aacc4930-6c55-41fa-a597-22ccdded594d
[18] MONK, ELLEN; BRET, Wagner. *Concepts in enterprise resorce planning* [online]. 2001. ISBN 9781111820398. Dostupné z: doi:10.1002/1521-3773(20010316)40:6<9823::AID-ANIE9823>3.3.CO;2-C
[19] RODRIGUEZ, Alex. RESTful Web Services. In: *Professional Java® EE Design Patterns* [online]. 2015 [vid. 2020-03-01], s. 165–182. Dostupné z: doi:10.1002/9781119209393.ch13
[20] ROY T. FIELDING. *rest-discuss : Message: Re: [rest-discuss] RFC for REST?* [online]. 2006 [vid. 2020-03-01]. Dostupné z: https://web.archive.org/web/20091111012314/http://tech.groups.yahoo.com/group/rest-discuss/message/6757
[21] ROY THOMAS FIELDING. Fielding Dissertation: CHAPTER 5: Representational State Transfer (REST). *Architectural Styles and the Design of Network-based Software Architectures*. 2000.

6 SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

Testbed – Přípravek pro zkoušení strojů nebo principů

C# - Programovací jazyk vyvinutý společností Microsoft s důrazem na objektově orientované programování

.NET Framework – Sada programovacích nástrojů vyvinutých společností Microsoft, úzce spjata s jazykem C#

REST - Representational state transfer, standard pro komunikaci mezi systémy založený na webových technologiích

SCADA - Supervisory control and data acquisition

PLC – Programmable logic controller

Serializace – převod objektu na text

Deserializace – převod textu na objekt

JSON – JavaScript object notation, formát pro přenos datových objektů ve formě textu

7 PŘÍLOHA

Projekt plánovače a všechna vstupní a výstupní data jsou zabalena v souboru „BP_Kubasek.zip“.

7.1 Vstupy

7.1.1 Machines


```
[
 {
  "Id": 5,
  "Job": 0,
  "JobEndsAt": null,
  "Name": "Fanuc",
  "State": 1
 },
 {
  "Id": 6,
  "Job": 0,
  "JobEndsAt": null,
  "Name": "Freza",
  "State": 1
 },
 {
  "Id": 7,
  "Job": 0,
  "JobEndsAt": null,
  "Name": "Kuka",
  "State": 1
 },
 {
  "Id": 8,
  "Job": 0,
  "JobEndsAt": null,
  "Name": "Transporter",
  "State": 1
 }
]
```

7.1.2 Operations


```
[
 {
  "Code": "",
  "Duration": 10,
  "Expression": "Tvydej\/Tfanuc",
  "Id": 12,
  "MachineId": 8,
  "Name": "Transport Vydej - Fanuc"
 },
 {
  "Code": "",
  "Duration": 10,
  "Expression": "Tfanuc\/Tvydej",
  "Id": 11,
  "MachineId": 8,
  "Name": "Transport Fanuc-Vydej"
 },
 {
  "Code": "",
  "Duration": 10,
  "Expression": "Tkuka\/Tvydej",
  "Id": 10,
  "MachineId": 8,
  "Name": "Transport Kuka-Vydej"
 },
 {
  "Code": "",
  "Duration": 10,
  "Expression": "Tkuka\/Tfanuc",
  "Id": 9,
  "MachineId": 8,
  "Name": "Transport Kuka-Fanuc"
 },
 {
  "Code": "",
  "Duration": 10,
  "Expression": "Tfanuc\/Tkuka",
  "Id": 8,
  "MachineId": 8,
  "Name": "Transport Fanuc-Kuka"
 },
 {
  "Code": "",
  "Duration": 3,
  "Expression": "Tkuka,kapsa\/Tkuka,privesek",
  "Id": 7,
```

```
  "MachineId": 7,
  "Name": "Kompletuj klicenku"
},
{
  "Code": "",
  "Duration": 5,
  "Expression": "Tvydej\/Tkuka",
  "Id": 20,
  "MachineId": 6,
  "Name": "Transport Vydej - Kuka"
},
{
  "Code": "",
  "Duration": 5,
  "Expression": "freza,hrub\/freza,hrub,pero",
  "Id": 18,
  "MachineId": 6,
  "Name": "Frezovani pera"
},
{
  "Code": "",
  "Duration": 5,
  "Expression": "freza,hrub\/freza,hrub,slot",
  "Id": 17,
  "MachineId": 6,
  "Name": "Frezovani slotu vizitka"
},
{
  "Code": "",
  "Duration": 10,
  "Expression": "freza,hrub\/freza,hrub,text",
  "Id": 16,
  "MachineId": 6,
  "Name": "Frezovani textu"
},
{
  "Code": "",
  "Duration": 5,
  "Expression": "freza,hrub,kapsahrube\/freza,hrub,kapsa",
  "Id": 15,
  "MachineId": 6,
  "Name": "Frezovani otvoru freza 2 - jemne"
},
{
  "Code": "",
  "Duration": 5,
  "Expression": "freza,hrub\/freza,hrub,kapsahrube",
```

```
  "Id": 14,
  "MachineId": 6,
  "Name": "Frezovani otvoru freza 10 - hrub"
},
{
  "Code": "",
  "Duration": 10,
  "Expression": "freza\/freza,hrub",
  "Id": 13,
  "MachineId": 6,
  "Name": "Hrubovani"
},
{
  "Code": "",
  "Duration": 5,
  "Expression": "fanuc\/sklad",
  "Id": 6,
  "MachineId": 5,
  "Name": "Vloz do skladu"
},
{
  "Code": "",
  "Duration": 5,
  "Expression": "Tfanuc\/fanuc",
  "Id": 5,
  "MachineId": 5,
  "Name": "Vyjmi z transportu"
},
{
  "Code": "",
  "Duration": 10,
  "Expression": "fanuc\/Tfanuc",
  "Id": 4,
  "MachineId": 5,
  "Name": "Vloz na transport"
},
{
  "Code": "",
  "Duration": 10,
  "Expression": "freza\/fanuc",
  "Id": 3,
  "MachineId": 5,
  "Name": "Vyjmi z frezy"
},
{
  "Code": "",
  "Duration": 5,
```

```
  "Expression": "fanuc\/freza",
  "Id": 2,
  "MachineId": 5,
  "Name": "Vloz do frezy"
 },
 {
  "Code": "",
  "Duration": 10,
  "Expression": "sklad\/fanuc",
  "Id": 1,
  "MachineId": 5,
  "Name": "Vyjmi ze skladu"
 }
]
```


7.1.3 Items


```
[
 {
  "Id": 3,
  "Labels": "fanuc",
  "OrderId": -1
 },
 {
  "Id": 4,
  "Labels": "sklad",
  "OrderId": -1
 }
]
```


7.1.4 Orders


```
[
 {
  "Id": 3,
  "Stav": 3,
  "Ukonceno": "2019-09-25T10:47:47Z",
  "Zadani": "Tvydej,text=aaa,hrub,slot,pero,privesek",
  "Zadano": "2019-09-25T10:47:38Z",
  "Zdroj": "APP"
 },
 {
  "Id": 2,
  "Stav": 0,
  "Ukonceno": null,
  "Zadani": "Tvydej,text=aaa,hrub,slot,pero,privesek",
  "Zadano": "2019-02-01T00:00:00Z",
```

```
  "Zdroj": null
 },
 {
  "Id": 1,
  "Stav": 0,
  "Ukonceno": null,
  "Zadani": "Tvydej,text=aaa,hrub,slot,pero,privesek",
  "Zadano": "2019-01-01T00:00:00Z",
  "Zdroj": null
 }
]
```


7.2 Výstupy

7.2.1 Activities


```
[
 {
  "DependentOn": -1,
  "EndTime": "2020-06-03T14:03:27Z",
  "Id": 1,
  "ItemId": 4,
  "OperationId": 1,
  "StartTime": "2020-06-03T13:53:27Z"
 },
 {
  "DependentOn": 1,
  "EndTime": "2020-06-03T14:08:27Z",
  "Id": 2,
  "ItemId": 4,
  "OperationId": 2,
  "StartTime": "2020-06-03T14:03:27Z"
 },
 {
  "DependentOn": -1,
  "EndTime": "2020-06-03T14:13:27Z",
  "Id": 3,
  "ItemId": 3,
  "OperationId": 2,
  "StartTime": "2020-06-03T14:08:27Z"
 },
 {
  "DependentOn": 15,
  "EndTime": "2020-06-03T14:58:27Z",
  "Id": 4,
```

```
  "ItemId": 4,
  "OperationId": 3,
  "StartTime": "2020-06-03T14:48:27Z"
},
{
  "DependentOn": 3,
  "EndTime": "2020-06-03T15:08:27Z",
  "Id": 5,
  "ItemId": 4,
  "OperationId": 4,
  "StartTime": "2020-06-03T14:58:27Z"
},
{
  "DependentOn": 15,
  "EndTime": "2020-06-03T15:38:27Z",
  "Id": 6,
  "ItemId": 3,
  "OperationId": 3,
  "StartTime": "2020-06-03T15:28:27Z"
},
{
  "DependentOn": 3,
  "EndTime": "2020-06-03T15:48:27Z",
  "Id": 7,
  "ItemId": 3,
  "OperationId": 4,
  "StartTime": "2020-06-03T15:38:27Z"
},
{
  "DependentOn": 2,
  "EndTime": "2020-06-03T14:18:27Z",
  "Id": 8,
  "ItemId": 4,
  "OperationId": 13,
  "StartTime": "2020-06-03T14:08:27Z"
},
{
  "DependentOn": 13,
  "EndTime": "2020-06-03T14:23:27Z",
  "Id": 9,
  "ItemId": 4,
  "OperationId": 18,
  "StartTime": "2020-06-03T14:18:27Z"
},
{
  "DependentOn": 18,
  "EndTime": "2020-06-03T14:28:27Z",
```

```
 "Id": 10,
 "ItemId": 4,
 "OperationId": 17,
 "StartTime": "2020-06-03T14:23:27Z"
},
{
 "DependentOn": 17,
 "EndTime": "2020-06-03T14:38:27Z",
 "Id": 11,
 "ItemId": 4,
 "OperationId": 16,
 "StartTime": "2020-06-03T14:28:27Z"
},
{
 "DependentOn": 16,
 "EndTime": "2020-06-03T14:43:27Z",
 "Id": 12,
 "ItemId": 4,
 "OperationId": 14,
 "StartTime": "2020-06-03T14:38:27Z"
},
{
 "DependentOn": 14,
 "EndTime": "2020-06-03T14:48:27Z",
 "Id": 13,
 "ItemId": 4,
 "OperationId": 15,
 "StartTime": "2020-06-03T14:43:27Z"
},
{
 "DependentOn": 2,
 "EndTime": "2020-06-03T14:58:27Z",
 "Id": 14,
 "ItemId": 3,
 "OperationId": 13,
 "StartTime": "2020-06-03T14:48:27Z"
},
{
 "DependentOn": 13,
 "EndTime": "2020-06-03T15:03:27Z",
 "Id": 15,
 "ItemId": 3,
 "OperationId": 18,
 "StartTime": "2020-06-03T14:58:27Z"
},
{
 "DependentOn": 18,
```

```
  "EndTime": "2020-06-03T15:08:27Z",
  "Id": 16,
  "ItemId": 3,
  "OperationId": 17,
  "StartTime": "2020-06-03T15:03:27Z"
 },
 {
  "DependentOn": 17,
  "EndTime": "2020-06-03T15:18:27Z",
  "Id": 17,
  "ItemId": 3,
  "OperationId": 16,
  "StartTime": "2020-06-03T15:08:27Z"
 },
 {
  "DependentOn": 16,
  "EndTime": "2020-06-03T15:23:27Z",
  "Id": 18,
  "ItemId": 3,
  "OperationId": 14,
  "StartTime": "2020-06-03T15:18:27Z"
 },
 {
  "DependentOn": 14,
  "EndTime": "2020-06-03T15:28:27Z",
  "Id": 19,
  "ItemId": 3,
  "OperationId": 15,
  "StartTime": "2020-06-03T15:23:27Z"
 },
 {
  "DependentOn": 8,
  "EndTime": "2020-06-03T15:21:27Z",
  "Id": 20,
  "ItemId": 4,
  "OperationId": 7,
  "StartTime": "2020-06-03T15:18:27Z"
 },
 {
  "DependentOn": 8,
  "EndTime": "2020-06-03T16:01:27Z",
  "Id": 21,
  "ItemId": 3,
  "OperationId": 7,
  "StartTime": "2020-06-03T15:58:27Z"
 },
 {
```

```
  "DependentOn": 4,
  "EndTime": "2020-06-03T15:18:27Z",
  "Id": 22,
  "ItemId": 4,
  "OperationId": 8,
  "StartTime": "2020-06-03T15:08:27Z"
 },
 {
  "DependentOn": 7,
  "EndTime": "2020-06-03T15:31:27Z",
  "Id": 23,
  "ItemId": 4,
  "OperationId": 10,
  "StartTime": "2020-06-03T15:21:27Z"
 },
 {
  "DependentOn": 4,
  "EndTime": "2020-06-03T15:58:27Z",
  "Id": 24,
  "ItemId": 3,
  "OperationId": 8,
  "StartTime": "2020-06-03T15:48:27Z"
 },
 {
  "DependentOn": 7,
  "EndTime": "2020-06-03T16:11:27Z",
  "Id": 25,
  "ItemId": 3,
  "OperationId": 10,
  "StartTime": "2020-06-03T16:01:27Z"
 }
]
```


7.2.2 Xlsx report

Dostupný pouze jako přiložený soubor „Report.xlsx“.