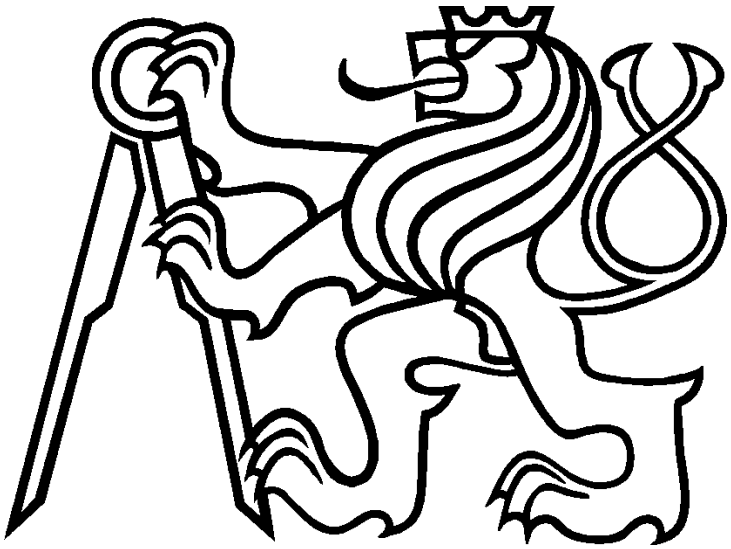
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA ELEKTROTECHNICKÁ

Katedra řídicí techniky



DIPLOMOVÁ PRÁCE

Simulace rozsáhlejších výrobních celků

2014 Tomáš URBAN

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze, dne 12.5.2014 .....

.....

podpis

Poděkování

Rád bych tímto poděkoval vedoucímu diplomové práce, Ing. Pavlu Burgetovi, PhD., a to za poskytnuté rady a odborné vedení. Dále děkuji rodině a mým nejbližším, kteří mne po celou dobu studia podporovali a bez nichž by teď pravděpodobně nebylo ani co číst.

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra řídicí techniky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Bc. Tomáš Urban**

Studijní program: Kybernetika a robotika
Obor: Systémy a řízení

Název tématu: **Simulace rozsáhlejších výrobních celků**

Pokyny pro vypracování:

1. Seznamte se s prostředím Plant Simulation a s možnostmi modelování spotřeby elektrické energie na linkách.
2. Navrhněte v Plant Simulation výrobní linku, která bude obsahovat nejméně 10 výrobních buněk, přičemž alespoň jedna z nich bude mít odpovídající model v Process Simulate. Prozkoumejte možnosti provázání modelu v Plant Simulation a Process Simulate.
3. Na celé výrobní lince optimalizujte výrobu vzhledem k různým hlediskům, jako je propustnost linky, velikost mezilehlých zásobníků, spotřeba elektrické energie apod.

Seznam odborné literatury:

Bangsow, S. Manufacturing Simulation with Plant Simulation and SimTalk. Springer 2010. ISBN 978-3-642-05073-2.
Process Simulate Reference Manual. Siemens Industry Software. 2013.

Vedoucí: Ing. Pavel Burget, Ph.D.

Platnost zadání: do konce letního semestru 2014/2015

prof. Ing. Michael Šebek, DrSc.
vedoucí katedry

L.S.

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 12. 2. 2014

ANOTACE

Tématem této práce je simulování výrobních procesů ve virtuálním prostředí. V dnešní době se k tomuto postupu přiklání stále více firem produkujících výrobně náročnější produkty. Počítačová simulace, kromě ověření samotného výrobního postupu, umožňuje optimální naladění výrobní linky bez současného zásahu do provozu.

Veškeré simulace jsou realizovány v programu Plant Simulation, jehož stručný popis je součástí této studie. Na reálném příkladu výrobní linky je demonstrována funkčnost programu. V prvé řadě jde o vytvoření samotného virtuálního obrazu části továrny, pozornost je však věnována i analýze linky a konkrétním experimentům zaměřeným především na úsporu energie.

ANNOTATION

The theme of this thesis is the simulations of productive processes in a virtual environment. More and more companies producing production-intensive products incline to this procedure in these days. Computer simulations, in addition to verifying the actual manufacturing process, enable optimal tuning of the production line without any impact on the real manufacturing process.

All simulations are carried out in the Plant Simulation, a brief description of this is also included in this study. The real example of the production line demonstrates functionality of the program. First of all it is about creating a virtual image of a part of the factory, attention is paid to the analysis of a production line and concrete optimization processes primarily focused on energy saving.

OBSAH

I SEZNAM OBRÁZKŮ

II SEZNAM TABULEK

1 ÚVOD DO SIMULOVÁNÍ VÝROBNÍCH PROCESŮ

Díky velkému rozvoji robotických systémů v posledních letech, které jsou stále univerzálnější a tak použitelnější pro daleko větší spektrum potenciálních výrobků, není robotizovaná výroba již doménou pouze dopravního průmyslu, ale ve stále větší míře ji můžeme vidět i v jiných odvětvích, například v potravinářství. Důvodů k přechodu na automatizovanou výrobu je několik, jedná se především o snížení nákladů na provoz, zefektivnění výroby či její samotné zrychlení.

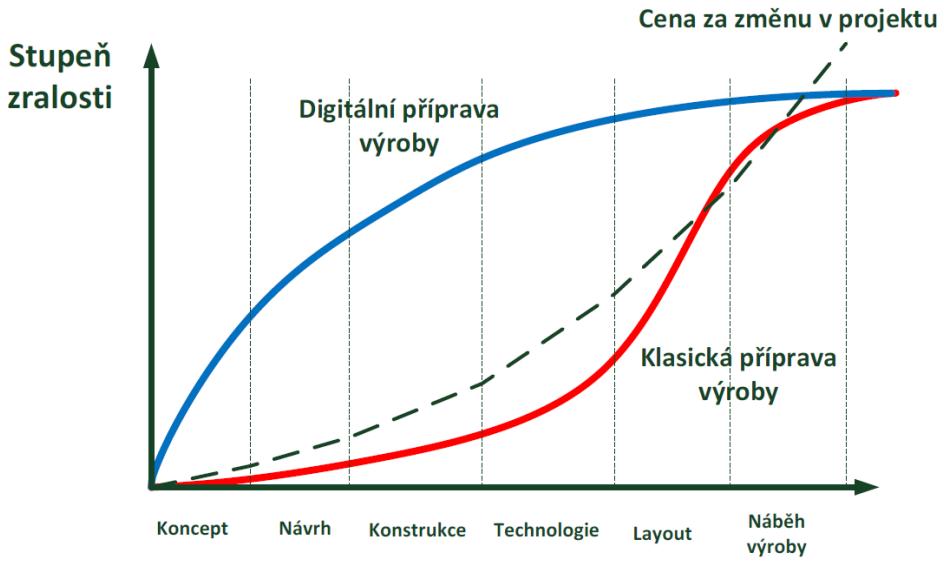
Problém často nastává se zprovozněním automatizované výroby a synchronizací jednotlivých výrobních procesů. Problémový je taktéž přechod na nový výrobek, kdy se celá linka musí přeprogramovat a odzkoušet. Celý proces může trvat i několik dní, výroba neběží a dochází tak k nemalým finančním ztrátám.

Zavádíme tak pojem „digitální továrna". Digitální továrna je virtuálním obrazem reálné výroby, který zobrazuje výrobní procesy ve virtuálním prostředí. Prostředí digitální továrny tak umožňuje simulovat celý výrobní proces nezávisle na dostupnosti fyzické výrobní linky. Po vyladění simulace je možno přímo ze simulačního prostředí vygenerovat program pro průmyslové roboty, díky tomu je rozjetí reálné linky možné v daleko kratším čase a v ideálním případě bez komplikací. Mezi další přednosti digitální továrny patří:

- plánování výroby
- optimalizace výroby
- zefektivnění výroby
- předpřipravení výrobní linky pro budoucí změnu výrobku

Pro co nejvyšší využití potenciálu digitální továrny je snaha o co největší integraci celé výroby. Komplexním a ideálním řešením by bylo propojení prostředí digitální továrny s fyzickými zařízeními linky přímo. Tato vlastnost však zatím není výrobci (jak hardwaru, tak softwaru) přímo podporována. Nicméně se začínají objevovat ucelené softwarové

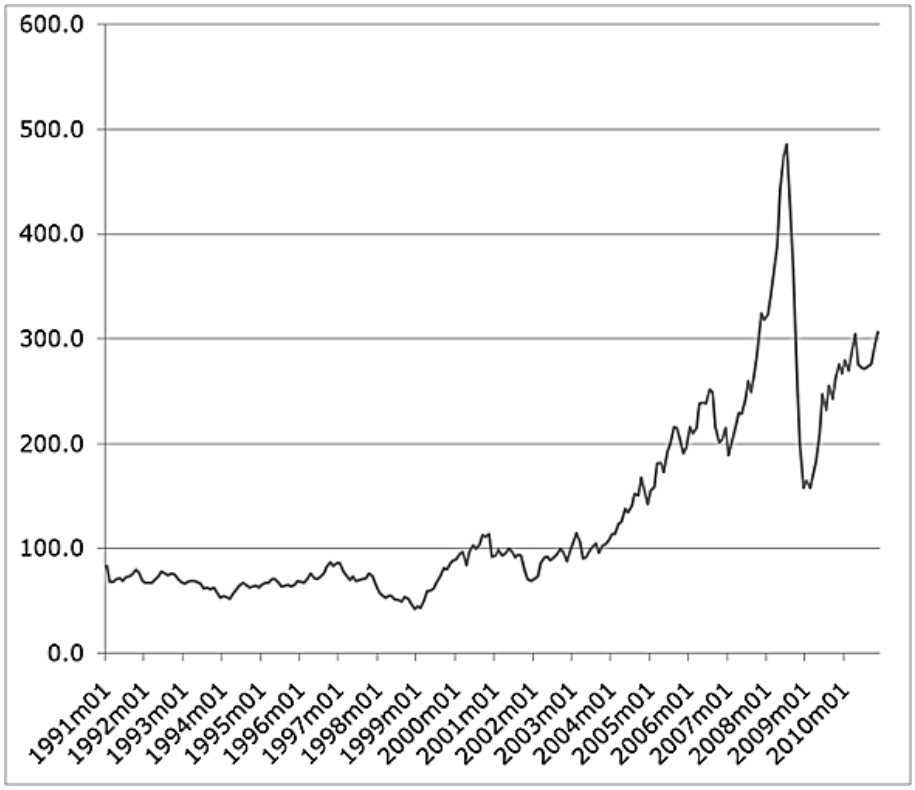
nástroje, které mají na starosti celý výrobní cyklus výrobku, od jeho návrhu, po realizaci, testování, plánování výroby i logistiky a následnou podporu. Tím dochází ke značnému zjednodušení, a to jak z hlediska práce s virtuálním obrazem výrobny, tak díky úzké spolupráci jednotlivých nástrojů i k následnému zefektivnění výroby.



Obr. 1.1 Rozdíl mezi klasickou a digitální přípravou výroby (zdroj [12])

Čemu je, především v poslední době, věnována velká pozornost, je úspora. Zde konkrétně máme na mysli úsporu energií. V rozsáhlých výrobních areálech, které čítají desítky až stovky robotických zařízení (například automobilová továrna), jsou napájena, mimo jiné, i zařízení, která se v danou chvíli nevyužívají. Zde pak existují dva směry, jakými lze úspory docílit. Prvním směrem je využití možnosti přepnutí do úsporného režimu u zařízení, která nejsou momentálně využívána. V dnešní době většina výrobních strojů tímto módem disponuje, málokdy je však v reálném provozu využíván. Druhou cestou k docílení úspory je optimalizovat celou sekvenci výrobních procesů tak, aby zařízení byla využívána po co nejdelší možnou dobu. Samozřejmě, v každé výrobě je snaha tohoto stavu dosáhnout, u velkých továren však není v lidských silách docílit maximální optimality bez dalších softwarových nástrojů. Výhodou tohoto přístupu je možnost kombinace několika kritérií pro optimalizaci. Nemusíme se tedy soustředit jen na energie,

ale i na využití lidských pracovních sil, na co nejnižší logistické náklady či na co nejkratší výrobní čas výrobku. Kritérií může být samozřejmě daleko více.



Obr. 1.2 Přehled vývoje ceny za energii (zdroj [13])

Tato práce kombinuje oba výše popsané přístupy a snaží se simulovat a jistým způsobem optimalizovat výrobu jak z pohledu továrny, to znamená, že bere konkrétní stanoviště jako jednu buňku o určitém procesním čase a konkrétních vlastnostech, tak i z hlediska jednotlivých stanic, které pracují v různých energetických módech a snahou je přepínat mezi těmito módy co nejvhodnějším způsobem. K analýze je využit nástroj Plant Simulation firmy Siemens, který je součástí balíku Tecnomatix.

1.1 Členění práce

První část se věnuje popisu softwarových nástrojů pro práci s digitální továrnou a pro virtuální zprovoznění výroby. Konkrétně se jedná o softwarové balíky Teamcenter a Tecnomatix. Pozornost je dále věnována simulacím obecně, především jejich využití, a postupům při návrhu.

Druhá část seznamuje s nástrojem Plant Simulation. Popsány jsou základní vlastnosti, ovládací prvky a předdefinované objekty pro modelování včetně jednoduchých ukázek použití.

V poslední části se již věnujeme konkrétnímu návrhu výrobní linky v simulačním prostředí. Celý postup je detailně popsán, a to včetně problémů a komplikací, které samotný návrh doprovázely. Vytvořený simulační model je pak podroben analýze a několika experimentům, které zkoumají vliv různých faktorů na konečnou produktivitu a efektivitu.

Práce je pojata jako jakýsi prostředek pro seznámení s programem Plant Simulation, který shrnuje základní fakta a ukazuje čtenáři, jakým způsobem se s ním pracuje. Jedním z cílů práce je možnost využití tohoto materiálu pro další výuku.

2 PLM SOFTWARE

2.1 Tecnomatix

Tecnomatix je softwarový balík pro virtuální zprovoznění výrobních a průmyslových linek. Umožňuje správu části životního cyklu výrobku (PLM) od digitálního plánování procesů, přes jejich ověřování, až po samotnou podporu výroby.

Obr. 2.1 Pokrytí životního cyklu výrobku softwarem Tecnomatix (vyznačené části)

Tecnomatix se skládá z několika softwarových nástrojů, mluvíme především o třech hlavních součástech pro plánování, simulování a optimalizování procesů v rámci digitální továrny.

2.1.1 Process Designer

Slouží k plánování výroby a procesů, rozmístění a prostorového umístění strojů v rámci výrobní haly, využívá se pro definování technologických postupů a plánování montáže výrobků. Proces výroby je popsán jako propojení základních datových typů, mezi které řadíme produkty, zdroje a operace. Process Designerem tedy stanovujeme, jakým zdrojem budeme jaký produkt a jak upravovat, včetně dalších informací, například, o časové normě procesu atd. Zdroj je zde myšlen jako výrobní prostředek, nejde tedy jen o konkrétní stroje, ale i o nářadí a samozřejmě i o pracovníky, jejichž práce se modeluje a plánuje také. V rámci Process Designeru se vytvoří detailní studie, která se importuje do Process Simulate pro následnou simulaci.

2.1.2 Process Simulate

Jedná se o softwarový nástroj určený pro různé inženýrské studie a simulace. Process Simulate přichází na řadu v momentě, kdy potřebujeme analyzovat a ověřovat koncept výroby ve fázi plánování, a to pouze za pomoci počítačového modelu. Je úzce propojen s Process Designerem a na základě definovaných zdrojů, produktů a operací dokáže doupravovat strukturu továrny a výrobní postupy, které následně simuluje. Umožňuje vytváření vnitřní logiky tak, aby byl ve výrobě zaručen správný postup. Díky simulaci můžeme ověřit proveditelnost montáže, analyzovat proveditelnost manuálních operací a s nimi spojené ergonomie, ověřovat lze taktéž PLC programy nebo například robotické programy pro svářecí roboty. Po odladění je možno vyexportovat robotické programy přímo pro konkrétní robotický kontrolér vybraného výrobce. V konečné fázi Process Simulate umožní modelovat i samotnou logistiku a balení hotových výrobků.

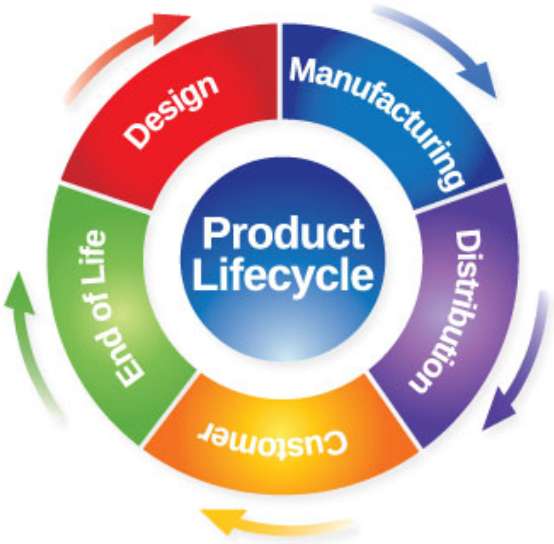
Jak již bylo popsáno, oba výše uvedené nástroje úzce spolupracují, a to díky serverové architektuře. Jejich využití se očekává ve velkých podnicích s rozsáhlou výrobou, kde bude několik stanic, na kterých bude paralelně pracovat několik uživatelů, sdílení dat je tak nutností. Konkrétně se jedná o 3-vrstvou architekturu s Oracle databází, kam se ukládají veškerá data, střední vrstvu tvoří eMServer, což je aplikační server obou nástrojů, a poslední vrstvu tvoří samotné klientské počítače s nainstalovanými programy Process Simulate a Process Designer.

2.1.3 Plant Simulation

Není již tak propojen s výše uvedenými programy, Plant Simulation bere jednotlivá pracoviště jako jeden celek a přímo se nesoustředí na konkrétní ovládání a řízení koncových prvků ve výrobě. Jeho úkolem je optimalizovat proces v rámci celé továrny, především při přechodu mezi pracovišti. Umožňuje vytvářet strategie, statistiky a experimenty. V tomto případě nejde o tolik technicky orientovaný nástroj, ale především o nástroj plánovací, který vidí výrobu o úroveň výše než nástroje Process Designer a Process Simulate. Jelikož je tento program pro tuto práci stěžejní, bude mu podrobnější pozornost věnována v následujících kapitolách.

2.2 Teamcenter

Teamcenter je momentálně nejrozšířenějším PLM softwarem na světě. Jedná se o komplexnější nástroj než balík Tecnomatix, na rozdíl od něj totiž spravuje celý životní cyklus výrobku, nejen výrobní a plánovací proces. Teamcenter je tak určen pro velké společnosti, které vyrábějí technicky složité výrobky, například automobilky. Poskytuje v rámci celé firmy přístup k produktovým a procesním znalostem, a umožňuje tak lépe pracovat v rámci celého životního cyklu výrobku.



Obr. 2.2 Teamcenter - správa nad celým životním cyklem výrobku

V principu můžeme konstatovat, že Teamcenter je nadstavbou softwarového balíku Tecnomatix, jehož funkce integruje a k tomu přidává množství dalších nástrojů pro návrh designu, management, administraci, podporu výrobku a podobně. Pokrývá tak všechny odvětví z obr. 2.1.

Řešení Teamcenter zahrnuje:

- systémové inženýrství a řízení požadavků
- správu portfolia, programů a projektů
- řízení konstrukčního procesu
- řízení kusovníků
- zajištění souladu s předpisy
- správu obsahu a dokumentů
- řízení vzorů, balení a značky
- řízení vztahů s dodavateli
- řízení mechatronických procesů
- řízení procesu výroby
- řízení simulačního procesu
- údržbu, servis a generální opravy
- reporty a analýzy
- týmovou spolupráci
- vizualizaci životního cyklu

Z výčtu vlastností je patrné, že tento nástroj vytváří jakousi znalostní základnu napříč celým podnikem. Nejde již jen o digitální továrnu, její verifikaci a plánování výrobních postupů, ale o celkový dohled nad firemními informacemi. Toto propojení značně zjednodušuje komunikaci mezi jednotlivými odděleními, ať už se jedná o výrobu, logistiku, návrh nebo samotné vedení firmy. Tím je zajištěna vyšší produktivita, zlepšení globální týmové práce, úspora finančních prostředků, rychlejší návratnost investic a kontrola celého životního cyklu výrobku.

Teamcenter je softwarový balík, nicméně jeho složení si zákazník volí sám. Například menší a střední firmy nemají potenciál k využití veškerých jeho součástí, proto by pro ně investice do celého balíku byla zbytečná. Protože Teamcenter ve své podstatě pokrývá stejnou oblast jako Tecnomatix, momentální strategií firmy je postupně přecházet z Tecnomatixu na Teamcenter s daleko větším využitím, uplatněním a případným rozšířením do budoucna. Prostředí je však u obou systémů velmi podobné, stejně tak jeho ovládání, čili by s přechodem, alespoň po technické stránce, neměl být problém.

3 SIMULACE VÝROBNÍCH POSTUPŮ

3.1 Oblasti použití

Simulaci můžeme podle situace využít ve fázi návrhu, v implementační fázi nebo ve fázi provozní. Nic však nebrání využití ve všech třech oblastech použití. V každé fázi nám simulace pomůže odpovědět na jiné otázky, jejich možná podoba je popsána v následujících podkapitolách.

3.1.1 Fáze návrhu

Zde simulace pomůže identifikovat „úzká" místa v procesu výroby, tedy například místa s malou nebo pomalou propustností materiálu. Můžeme také nalézt skrytý a nevyužitý potenciál některých částí a změnou parametrů docílit vyšší efektivity linky. Simulace nám dokáže vzájemně porovnat různé plánovací alternativy výroby, stejně tak produkci výrobků za různých podmínek tak, aby bylo nakonec dosaženo maximálního využití linky. Ve fázi návrhu dokážeme předem stanovit potřebnou kapacitu skladů a zásobníků, počet pracovníků a výrobních zařízení, a v neposlední řadě také přibližnou spotřebu energie, která musí být u větších podniků známa předem. Odzkoušet lze i náročnost přechodu na nový výrobek, výkonnostní limity linky, logistickou dostupnost nebo dopředu analyzovat případné překážky, které se za běhu mohou objevit.

3.1.2 Implementační fáze

Při samotné realizaci můžeme provádět výkonnostní testy, analyzovat již konkrétní problémy a zkoumat jejich dopad na průběh výroby. Testujeme zde i odezvu systému na budoucí požadavky a porovnáváme efektivitu za různých podmínek, ať už plánovaných nebo neplánovaných. V této fázi již máme k dispozici informace o poruchovosti zařízeních použitých ve výrobě, máme přesné procesní časy jednotlivých pracovišť a víme počet zaměstnanců, které můžeme využít. Simulace se díky těmto informacím zpřesní a dokáže velmi dobře kopírovat reálný proces. Kromě již uvedeného, v realizační fázi dochází

k optimalizaci výroby vzhledem k požadovaným kritériím, a to jak momentálním, tak dlouhodobým.

3.1.3 Provozní fáze

Za samotného provozu simulaci použijeme při stanovení dodacích lhůt odběratelům, stejně jako požadavků na dodávky materiálu od dodavatelů. Můžeme analyzovat nouzové a nové strategie, testovat řídicí alternativy a plánovat budoucí výrobu.

3.2 Proces návrhu simulace

Tato kapitola vysvětluje přibližný postup, kterého bychom se měli při návrhu simulace držet.

3.2.1 Formulace problémů

V první řadě je potřeba stanovit požadavky simulace. Tedy co přesně budeme simulovat, jakým způsobem a na jaké problémy se budeme soustředit.

3.2.2 Ověření simulace

V rané fázi návrhu je třeba ověřit možnosti samotné simulace na základě zadaných specifikací. Je třeba se soustředit především na dodaná data a informace o výrobní lince, zda existují vhodné matematické modely pro převod do virtuálního prostředí. Zajímá nás přesnost údajů, složitost výroby, počet proměnných, návaznost systémů a podobně. Po zvážení všech základních faktorů můžeme přistoupit k dalšímu kroku.

3.2.3 Definice cílů

Jedná se o stěžejní část, kvůli které se celá studie vytváří a jejíž výsledek nás zajímá nejvíce. Obvykle se stanovuje hlavní cíl, kterým bývá například ziskovost nebo třeba efektivita. Hlavní cíl se následně rozdělí na dílčí cílové položky, které spolu navzájem souvisí. Za časté cíle pro simulaci dále označujeme:

- minimalizace doby zpracování
- maximalizace využití (efektivita)
- minimalizace zásob
- plnění dodacích lhůt

Data pro zkoumání každé položky ze seznamu dílčích cílů mohou vyžadovat vždy trochu odlišné specifikace. Je tedy žádoucí mít tato data k dispozici, aby mohl být výsledek správně, anebo vůbec, analyzován.

3.2.4 Data

Následuje shromáždění samotných dat potřebných pro simulaci. Ty můžeme dělit na údaje o zatížení systému, organizační údaje a technické údaje. Mezi data o zatížení systému patří například pracovní plány, kusovníky, termíny a objednávky. Organizačními údaji máme na mysli pracovní dobu, směny, přestávky, výrobní zdroje a podobné. Technická data jsou především informace o rozmístění výrobních zdrojů, dopravní trasy, údaje o výkonu, kapacitě atd.

3.2.5 Modelování

V tomto kroku přistupujeme již k samotnému modelování. To můžeme rozdělit na dvě části. Nejprve je třeba vytvořit schématický - ikonický model z modelu konceptuálního. Vezmeme náš systém, který chceme simulovat, a na základě cílů, které budeme testovat, stanovíme požadavek na přesnost simulace. V této fázi dochází nejprve k analýze systému, tedy rozčlenění na jednotlivé subjekty, a následně k abstrakci tak, aby došlo k co

největšímu zjednodušení, avšak toto zjednodušení nesmí mít zároveň vliv na zkreslení výsledků simulace. Typickým zjednodušením je redukce (odstraníme nepodstatné detaily) nebo generalizace (zobecnění základních informací).

Druhou částí rozumíme převedení tohoto schématického modelu do simulačního softwaru. Model testujeme, zda se shoduje s původním vzorovým modelem a zda stanovená míra abstrakce nemá vliv na nepřesnosti. V případě, že simulace neodpovídá reálnému systému, je potřeba opakovat proces analýzy a abstrakce tak, aby se modely chovaly stejně. Je třeba myslet na to, že pokud v budoucnu budeme chtít linku optimalizovat podle jiných kritérií, náš virtuální model již nemusí vyhovovat.

3.2.6 Spuštění simulace

Na základě stanoveného cíle se spustí simulační studie a realizují se různé experimenty. Experimenty se liší ve vstupních datech, parametrech modelu a jeho objektů tak, aby se docílilo optimálních výsledků. Důležité je taktéž stanovit čas, po který bude experiment ve virtuálním prostředí běžet. Pokud zvolíme čas příliš krátký, některé proměnné parametry modelu se nemusí (nestačí) projevit. Příliš dlouhý čas poskytuje sice dobré statistické výsledky, nicméně zbytečně zatěžuje počítač a simulace experimentů rozsáhlejších projektů pak bývá časově velmi náročná. Vstupní a výstupní data každého experimentu jsou zaznamenávána.

3.2.7 Výsledky analýzy a jejich interpretace

Výsledek simulace by nám měl poskytnout parametry, které byly u modelovaného systému nalezeny nebo změněny, aby bylo dosaženo požadovaných cílů. Aby studie byla úspěšná, je důležité výsledky správně interpretovat. Může se stát, že dosažené hodnoty jsou v rozporu s předpoklady. Pak je nutno analyzovat vlivy, které jsou za tyto neočekávané výsledky odpovědné, a navrhnout experiment tak, aby výsledek odpovídal předpokladům.

Pro simulování komplexnějších a složitějších procesů výroby se doporučuje ze samotné simulace vynechat fázi „nájezdu" výroby. V této fázi model často neodpovídá realitě a do konečných statistik jsou tak přenášeny nepřesnosti.

3.2.8 Dokumentace

Nedílnou součástí celého procesu návrhu simulace je přehledná a detailní dokumentace. Dokumentace by měla vysvětlovat postupy, které byly při návrhu simulace použity. Stejně tak by zde měly být zaznamenány chybné varianty při vytváření virtuálního modelu. Stěžejní částí jsou samozřejmě dosažené výsledky. Dokumentace je důležitá jak pro shrnutí samotných výsledků simulace, tak pro případné další simulace, které pak nebude nutno vytvářet od počáteční fáze návrhu.

4 PLANT SIMULATION

Jak již bylo uvedeno, Plant Simulation je součástí balíku Tecnomatix vyvinutým společností Siemens PLM. Jedná se o software, který umožňuje modelování výrobních systémů, procesů, toku materiálu a logistických operací. Takto vytvořený virtuální model lze následně simulovat, analyzovat, vizualizovat, vytvářet statistiky provozu a optimalizovat podle požadovaných kritérií.

Plant Simulation je součástí PLM (Product Lifecycle Management Software), ale je zároveň samostatně fungujícím celkem. Díky tomu je dostupný i menším podnikům, kterým se nevyplatí (ať už finančně, nebo složitostí výrobní procedury) používat ostatní software z balíku Tecnomatix či Teamcenter.

Jedná se o takzvaný DES Software (Discrete Event Simulation Software), analýza všech procesů tak probíhá v diskrétním čase. Díky počítačovému modelu můžeme vytvářet různé scénáře a vybrat potom ten, který nejlépe vyhovuje našim kritériím a který je pro nás v jistém smyslu optimální. Reálnou výrobu pak (jen) upravíme podle simulace a výrobní proces může běžet dál, oproti optimalizaci přímo na reálném pracovišti ušetříme mnoho finančních prostředků a času. Plant Simulation je orientován na již běžící výrobu a její vyladění, nicméně můžeme ho stejně tak využít ještě před samotnou stavbou výrobní haly.

Klíčové vlastnosti:

- objektově orientované modely s hierarchií a odkazy
- otevřená architektura
- správa knihoven a objektů
- optimalizace pomocí genetických algoritmů
- simulace a analýza spotřeby energie
- automatická analýza výsledků simulace
- tvůrce HTML sestav
- 2D a 3D prezentace
- jazyk SimTalk pro tvorbu vlastních metod

Udávané hlavní přínosy:

- až 6% úspory při počáteční investici
- zvýšení produktivity stávajícího systému až o 20%
- snížení nákladů na nový systém až 20%
- optimalizace spotřeby prostředků a opakované použití
- snížení zásob až o 60%
- zkrácení doby propustnosti až o 60%
- optimalizace systémů pro snížení spotřeby energie

Konkrétní čísla u přínosů jsou samozřejmě pouze orientační, záleží na daném výrobním systému nebo procesu a na tom, jak dobře je před simulací navržen.

4.1 Grafické prostředí v Plant Simulationu

Po otevření nebo založení projektu se v základním zobrazení objeví okno zobrazené na obr. 4.1. Hlavní okno můžeme rozdělit na několik částí, kde každá má svoji funkci.

4.1.1 Horní menu

V horní části obrazovky se nachází klasické windows menu pro práci s projektem, nastavení zobrazení, nástrojů, modelu a programu jako takového. Nachází se zde i nápověda. Nejběžnější funkce jsou přístupné v podobě ikonek pod roletovým menu.

4.1.2 Toolbox

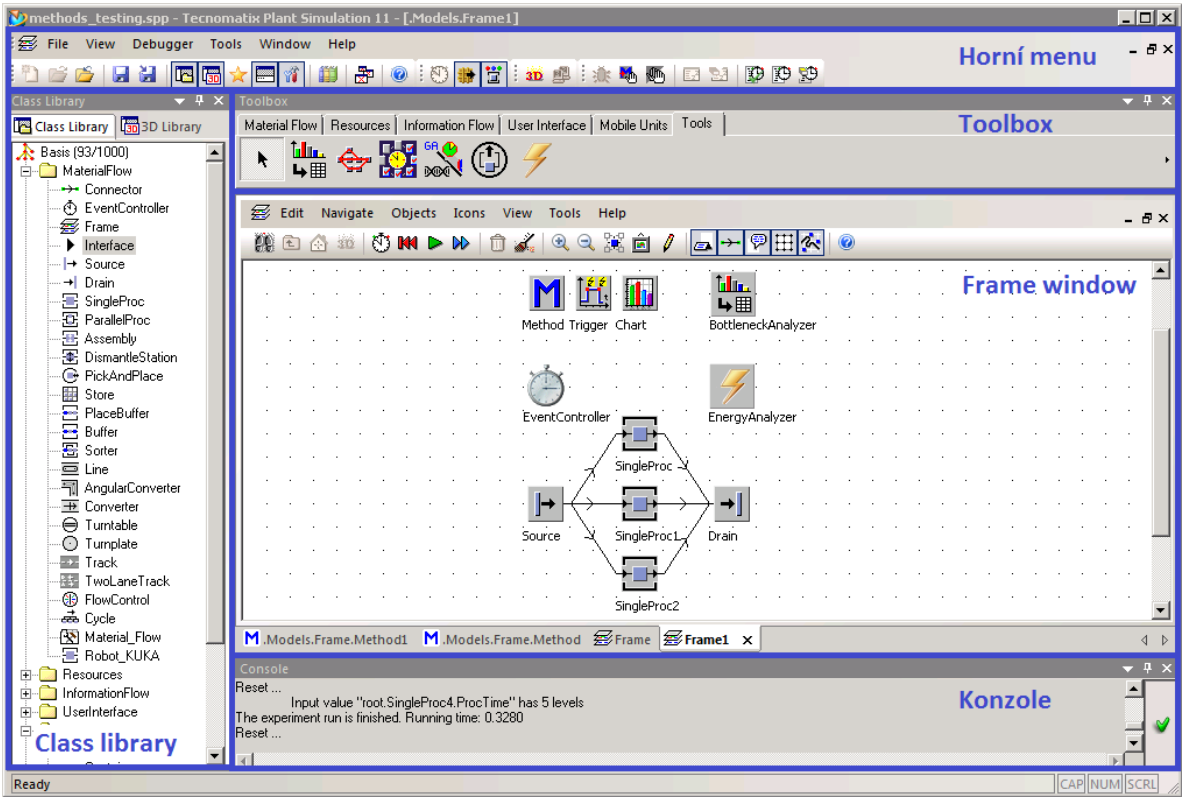
Toolbox umožňuje přístup do class library, pro přehlednost a jednoduché vkládání nabízí objekty v grafické podobě. Ty jsou rozděleny do záložek dle zaměření a funkce. Objekty můžeme přidávat i odebírat dle libosti, stejně tak si můžeme vytvořit novou záložku, například s námi vytvořenými novými objekty.

4.1.3 Frame window

Jedná se o okno pro vkládání objektů a vytváření schématického modelu pro simulaci. V horní části framu jsou ikony pro nejběžnější funkce, jako je ovládání simulace, nastavení zobrazení nebo přepnutí do grafického režimu.

4.1.4 Class library

Class library je strukturovaný adresář, kde nalezneme všechny objekty, které jsou pro simulaci potřeba. Struktura je otevřená a přístupná, lze ji tedy modifikovat dle našich požadavků.



Obr. 4.1 Hlavní okno Plant Simulationu

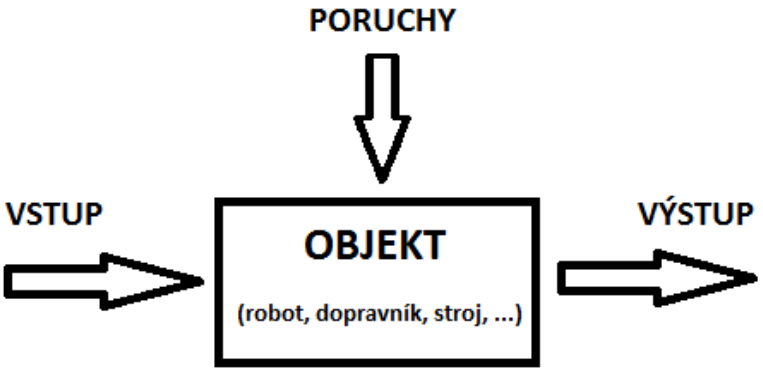
4.1.5 Konzole

Umožňuje zobrazování informací během simulace, například chybových hlášek. Konzoli je však možno využít i pro vypisování stavových a kontrolních informací, především pokud vytváříme vlastní metody.

Veškeré výše popsané grafické rozložení je defaultní a lze uživatelsky přizpůsobit - jednotlivé podokna je možno odebrat, přesunout, změnit jejich velikost nebo přidat další. Stejně tak můžeme měnit rozložení ikonických menu podle potřeby, Plant Simulation nabízí relativně vysokou míru customizace.

4.2 Modelování

Jak již bylo uvedeno, Plant Simulation je objektově orientovaný softwarový nástroj. Proces výroby je modelován za pomoci objektů, které znázorňují celé stroje, jejich části, pracovníky, dopravníky a další zařízení. K tomuto modelování je k dispozici pouze omezený počet základních objektů. Pokud reálný systém není možno popsat některým z definovaných objektů, je potřeba ho dále dekomponovat. Tak docílíme zjednodušení na konečné subsystémy, které je možno modelovat základními prvky programu. Tímto postupem dostaneme hierarchický model systému s top-down strukturou. Jednotlivé objekty a operace se pak spojují v celý výrobní proces. Určitý funkční celek systému vytváříme v rámci jednoho framu.



Obr. 4.2 Obecný popis objektu

4.2.1 Adresace

Hierarchická struktura umožňuje jednoduché a přesné adresování za pomoci tečkové notace. Například pro průmyslového manipulátora – robota KUKA_4 s adresou *tovarna.svarovna.hala3.pracoviste1.KUKA_4* na první pohled vidíme jeho umístění vzhledem k celé továrně.

Každý objekt specifikují jeho vlastnosti. U příkladu robota můžeme mluvit o jeho rychlostech, spotřebách, procesních časech, poruchách a o množství dalších specifikací. Tyto vlastnosti se nazývají atributy. Každý atribut má svůj název (attribute type) a hodnotu (attribute value). K atributům přistupujeme stejným způsobem jako k objektům a zahrnujeme je na konec adresy. Konkrétní případ pro našeho robota, jehož atributu *EnergyTargetState* přiřazujeme hodnotu typu string pro vypnutí:

tovarna.svarovna.hala3.pracoviste1.KUKA_4.EnergyTargetState := "Off"

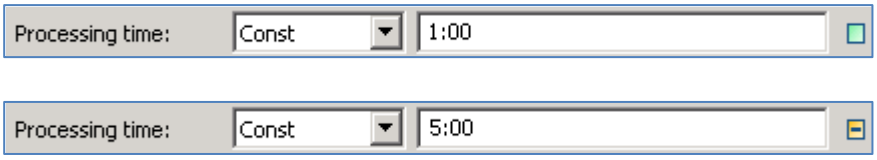
4.2.2 Třídy

Základním prvkem objektově orientovaných programovacích jazyků je třída (class). Umožňuje definování a vytváření vlastních objektů o určitých vlastnostech a určitém chování. Stejně je tomu i v případě Plant Simulationu. Pokud chceme použít nový objekt, bez návaznosti na již existující objekty, vytvoříme pro něj novou třídu. V té ho nadefinujeme a dostaneme tak zcela nový datový typ. V samotné simulaci poté využíváme jednotlivé instance této naší třídy, což jsou již konkrétní objekty, které mají stejné základní vlastnosti (atributy) jako třída a některé specifické vlastnosti instance (například jméno). Pro příklad můžeme uvést vytvoření nového typu stroje pro tvarování plastů. Vytvoříme pro něj třídu s jeho charakteristikou, tím dostaneme abstraktní objekt. V naší továrně jsou tyto stroje potřeba 3, vytvoříme tedy 3 instance, které jsou již reálnými objekty továrny.

4.2.3 Dědičnost

Další možností, jak vytvořit novou třídu, je využití dědičnosti. Děděním dostaneme podtřídu (subclass) odvozenou od již existující třídy. Veškeré vlastnosti jsou převzaty, avšak uživatel je může změnit podle potřeb, na původní třídu tyto změny již nebudou mít vliv. Výhodu tohoto přístupu oceníme při tvorbě objektů s mírně odlišnými vlastnostmi, než mají objekty, které máme k dispozici. Třídu tak není potřeba vytvářet od základu.

V samotném programu máme k dispozici 2 volby, a to duplicate a derive. Duplicate vytvoří kopii třídy, nová třída převezme i všechny vlastnosti, ale dále zde již neexistuje žádné propojení mezi originálem a duplikátem. Naopak derive vytvoří instanci z originální třídy, která může být objektem nebo novou podtřídou. Zde funguje stejné propojení jako u třídy a její instance, tedy změna vlastností ve třídě vyvolá změnu vlastností u instance. Tuto dědičnost lze u jednotlivých atributů vypnout, jak u třídy, tak u objektů. Děje se tak automaticky při změně atributů ve zděděném objektu, nebo můžeme dědičnost zrušit ručně zaškrtnutím políčka u jednotlivých vlastností.



Obr. 4.3 Horní obrázek – děděný atribut, dolní – dědičnost zrušena

Dědičnost je stěžejní vlastností programu a práce s objekty je na této funkcionalitě postavena.

4.3 Objekty - třídy

Standardní třídy (objekty) můžeme dělit do šesti kategorií:

- material flow objects
- resources
- information flow objects
- user interface/statistics objects
- mobile units
- general objects

Na úvod této podkapitoly je nutno upozornit, že následující objekty patří mezi základní a nejčastěji využívané. Některé z nich se v defaultním zobrazení nemusí v Toolboxu vyskytovat a naopak, popis některých zde bude chybět. Pokud se některý z objektů nezobrazuje, pravděpodobně nejsou v právě otevřeném modelu vybrány potřebné knihovny. Přes File – Manage Class Library můžeme kdykoli knihovny do našeho modelu přidat nebo je odebrat.

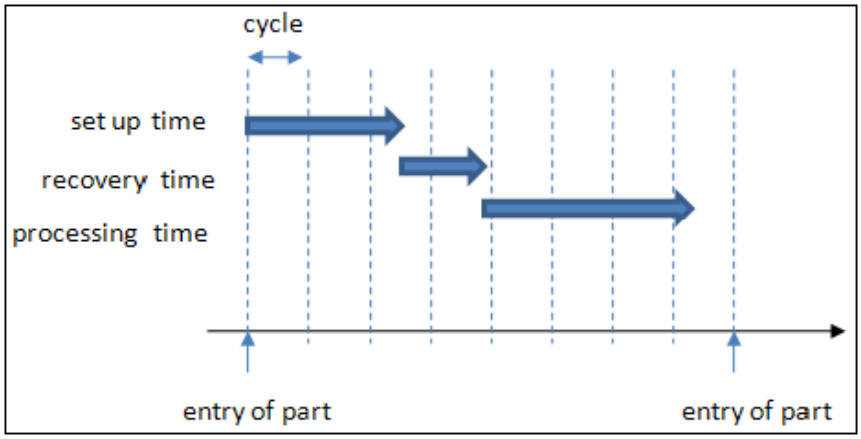
4.3.1 Material flow objects

Objekty materiálového toku jsou pro celou simulaci stěžejní. Můžeme je rozdělit do dvou skupin. Do té první patří mobilní jednotky (mobile units - MU), které symbolizují materiál nebo výrobek, který se opracovává a prochází tak postupně modelem linky nebo jeho částí. Druhou skupinou jsou statické objekty, které pracují s MU, přemisťují je, zpracovávají, obecně zajišťují jejich průchod skrz model. Pro větší přehlednost a dodržení terminologie, kterou Plant Simulation používá, jsou za material flow object v této podkapitole považovány právě tyto statické objekty. Mobilním objektům je pak věnována podkapitola vlastní.

4.3.1.1 Základní vlastnosti

Statické objekty můžeme nadále dělit na aktivní a pasivní. Aktivní objekty se podílí na samotném přesunu MU, jakmile s nimi dokončí určitou operaci, automaticky je odešlou v modelu dále. Jedná se například o roboty, stroje a dopravníky. Pasivní objekty s mobilními objekty taktéž pracují, ale sami je nedokážou poslat dále. Je to dáno jejich fyzikální podstatou. Uveďme například sklad nebo dráhu pro nějaký druh transportéru.

Plant Simulation pracuje s diskrétními událostmi. U objektů nastavujeme čas cyklu, který určuje, v jakých intervalech je možno přijmout další MU. Tento čas je důležitý pro synchronizaci přechodů mobilních jednotek mezi statickými objekty. Aby k přechodu mohlo dojít, musí být následující objekt volný, nesmí být v poruše nebo pozastaven a zároveň musí být splněna podmínka na celistvý násobek doby cyklu.



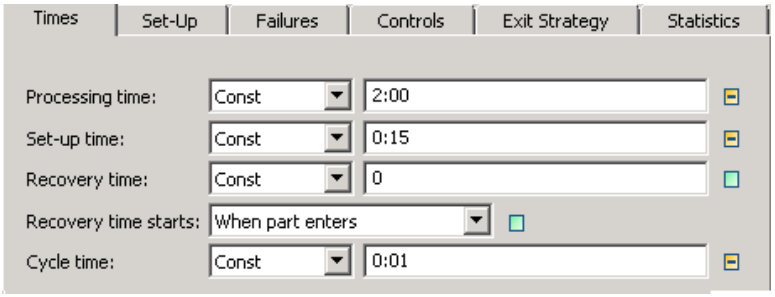
Obr. 4.4 Časové atributy objektů (zdroj [1])

Samotný průběh operace je pak možno rozdělit do tří částí:

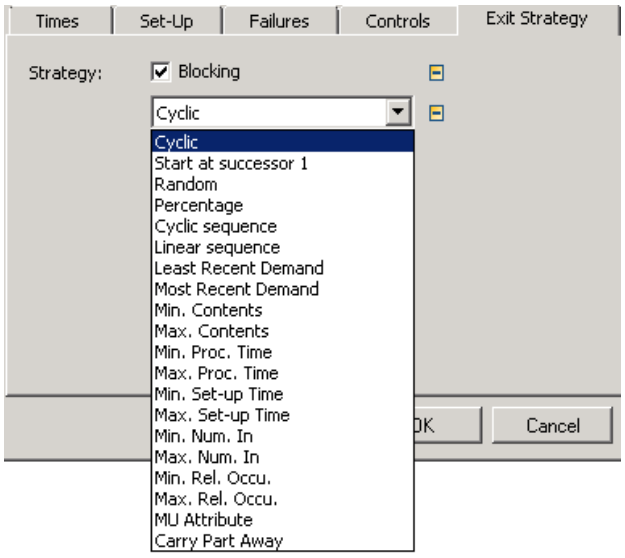
Setup time slouží pro definování času, který je potřeba pro přenastavení objektu/stroje v závislosti na typu příchozího MU. Pokud je typ odlišný od předchozího (jiný výrobek) může být potřeba například vyměnit nástroj u robota, což určitý čas trvá. Další možnost použití času nastavení je pravidelná údržba po určitém počtu zpracovaných MU - typicky například broušení čepiček u svařovacích kleští robota.

Recovery time využijeme, pokud po příchodu MU potřebujeme rezervovat určitý čas před započetím samotného procesu. Tímto způsobem modelujeme například čas potřebný pro založení dílu do stroje.

Processing time je čas samotné operace, kterou daný objekt představuje. Po setup time a recovery time jde o čas, po který je MU držen v objektu. Po uplynutí této doby je uvolněn a může se tak přesunout do objektu, který následuje.



Obr. 4.5 Nastavení časových atributů objektu



Obr. 4.6 Výběr výstupní strategie

4.3.1.2 Kapacita a blokování

Některé z objektů mají nastavitelnou kapacitu, která stanovuje, kolik MU může v jeden okamžik obsahovat. Rozměry mobilních objektů se běžně neberou v potaz, uvažujeme je pouze u objektů typu dopravník nebo třeba otočný stůl, kde se maximální kapacita naopak stanovuje právě na základě zadaných rozměrů.

MU, které z nějakého důvodu není přijato dalším objektem, je blokováno. Důvodem může být obsazenost následujícího objektu, jeho porucha, pozastavení nebo konec pracovní doby. Protože v modelu může následovat více potenciálních objektů, do kterých se MU může přesunout, existuje zde možnost volit výstupní strategii. Ta nabízí množství atributů pro výběr, podle kterých se rozhoduje, kam přesně se MU pošle a zdali se má v případě obsazenosti na tomto požadavku trvat (blokování), nebo se pro přesun může zvolit jiný volný objekt. Detailní popis výstupních strategií je například v [2].

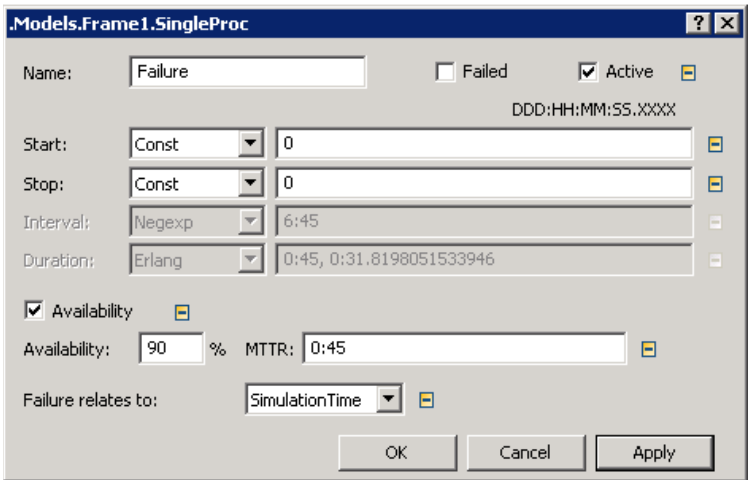
4.3.1.3 Poruchy

Kromě obsazenosti se následující objekt může vyskytnout v poruše. V tomto případě se probíhající proces objektu pozastaví, nové MU se nepřijímají, ale dokončené se ještě odešlou. Tímto způsobem se simuluje porucha stroje. Po odstranění poruchy se proces rozbíhá z bodu, ve kterém byl pozastaven. Objekt může být pozastaven i ručně bez nutnosti poruchy. Stejný efekt pozastavení má i přechod stroje do režimu Unplanned (mimosměnný provoz). Ručně je možno blokovat i samotný výstup nebo vstup objektu.

Základní stavy, jako je porucha, blokování, probíhající proces a podobné, jsou navíc indikovány barevnou kontrolkou v horní části objektu. V terminologii Plant Simulationu se tyto kontrolky nazývají LED.

Definování poruch je důležité pro co nejrealističtější simulaci. V rámci jednoho objektu je možno přednastavit několik na sobě nezávislých poruch, a to buď s přesným časovým určením, například pro pravidelnou údržbu stroje, nebo náhodně pro poruchy jako takové. Zde je pak na výběr mezi použitím pravděpodobnostního rozdělení se zadanými parametry nebo stanovením poruchy za pomoci Mean Time To Repair (MTTR) a availability (dostupnosti). V případě popisu konkrétními časy zadáváme časy, od kdy do kdy se porucha může projevit, interval s jakým se porucha vyskytuje a dobu samotné poruchy, pokud nastane. Kromě konstantních časů lze vybrat z množství statistických

rozdělení a zadat jejich parametry, více o tom například v [2]. Zadané časy mohou být vztaženy buď k průběhu celé simulace, to znamená, že nezáleží na stavu konkrétního objektu, ke kterému se porucha vztahuje, nebo přímo k procesnímu, popřípadě operačnímu, času tohoto objektu.



Obr. 4.7 Nastavení parametrů poruchy

Obr. 4.8 Grafické znázornění časů poruchy (zdroj [1])

Jak již bylo zmíněno, další možností specifikace poruchy je přes procentuální dostupnost systému a MTTR, tedy střední dobu do opravy. Po zadání těchto údajů program automaticky dopočítá střední dobu mezi poruchami (MTBF), platí totiž:

$$Availability = MTBF/(MTBF+MTTR)$$

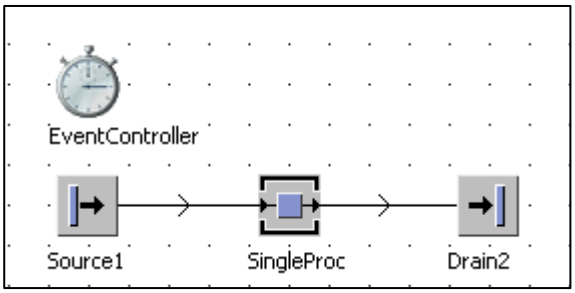
Doba trvání poruchy a interval mezi poruchami budou stanoveny vhodným pravděpodobnostním rozdělením, jehož parametry Plant Simulation automaticky dopočítá podle zadaných hodnot dostupnosti a MTTR.

4.3.1.4 Source

Produkuje mobilní objekty a tvoří tak počáteční hranici modelu. Objekt *Source* může vytvářet různé typy MU, přesné parametry jejich produkce zadáváme ve vlastnostech. Časy vytváření je možno stanovit pevně nebo opět za pomoci statistických rozdělení. Pokud zdroj vyrábí více druhů MU, jejich typy, počty a časy produkce definujeme v *Delivery Table*, kterou ke zdroji následně připojíme. Dále je možno stanovit, jakým způsobem bude MU z tabulky vybrán - náhodně nebo s určitou posloupností výběru.

4.3.1.5 Drain

V modelu plní opačnou funkci než *Source* a na konci, po zpracování všech simulovaných operací, maže MU. I *Drain* disponuje možností zadání poruch a procesních časů. Navíc shromažďuje množství statistických údajů o produkci, transportu a uložení materiálu. Tyto statistiky je dále možno využívat pro vytváření přehledů, grafů a optimalizačních strategií.



Obr. 4.9 Jednoduchý model se základními objekty

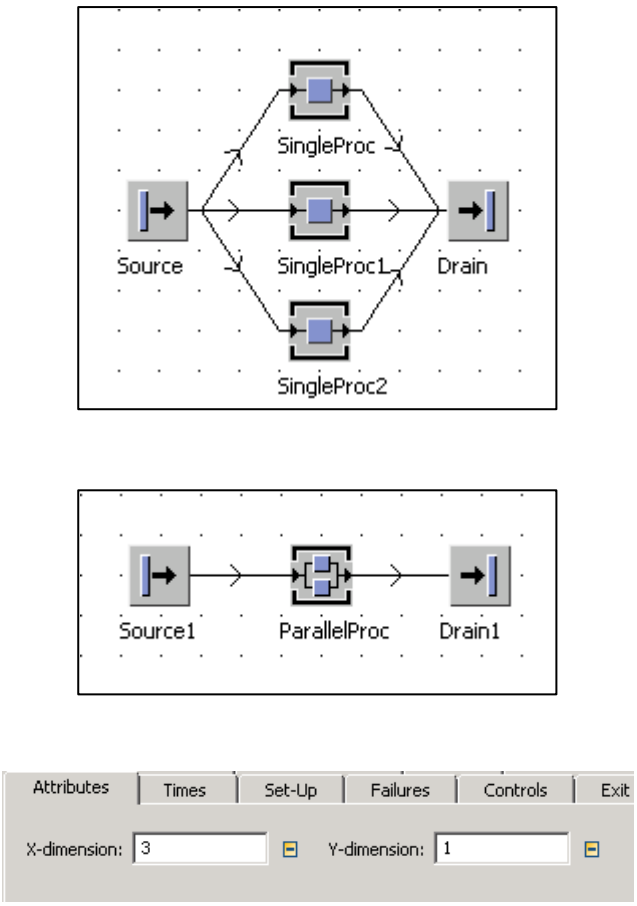
4.3.1.6 SingleProc

Představuje základní procesní jednotku jako určitou obecnou operaci s materiálem. Objekt simuluje jak práci stroje, tak například práci člověka, který však u stanice musí být přítomen. Podmínkou je zpracovávání pouze jednoho MU v jednom časovém okamžiku. Pokud je *SingleProc* volný, může přijmout materiál od předchozího objektu (předchůdce),

dále následuje samotná operace, která je určena setup, recovery a process časem a případně dalšími atributy jako například MTTR, availability, parametry spotřeby, řízení a obsluhy. Po dokončení je MU poslán k následujícím objektům (následníkům). Následníků může být i více, pak se volí výstupní strategie, která stanoví, kam přesně bude MU odeslán.

4.3.1.7 ParallelProc

Jedná se o vícenásobný *SingleProc* v jednom objektu. Pokud máme několik stejných paralelních pracovišť, je lepší využít právě *ParallelProc* - bez jakéhokoli řízení je MU přesunut na jednotku, která je momentálně nejdéle bez využití. Nastavení časů je možno zadat do tabulky pro každou jednotku zvlášť. V každé paralelní větvi (v programu atribut x-dimension) je možno vytvořit více objektů spojených do série (y-dimension).



Obr. 4.10 Stejná struktura modelována objekty SingleProc a ParallelProc

4.3.1.8 AssemblyStation

Slouží k modelování procesu montáže více MU, kdy je výsledkem jejich spojení nebo vytvoření nového MU. Pokud například vyrábíme stůl, jde o spojení jednoho MU typu deska a čtyř MU typu noha stolu. Výsledkem je zcela nový MU typu stůl nebo MU typu deska stolu se čtyřmi připojenými nohami. Co a jakým způsobem se má k čemu připojit, stanovujeme buď na základě čísla předchůdce (predecessor), odkud materiál přichází, nebo na základě typu MU. Vždy je třeba jednu z větví předchůdce zvolit za hlavní, po níž pak přichází i hlavní MU, ke kterému se ostatní části připojují.

4.3.1.9 Buffery

Rozlišujeme *PlaceBuffer* a *Buffer*. Přes *PlaceBuffer* MU prochází postupně jeden po druhém, jeho kapacita nám říká, kolik pozic je nutno projít, než bude MU uvolněn a poslán pryč. Jakmile je odeslán, veškeré mobilní objekty se ve struktuře posunují o jednu pozici dále. Čas procesu se zadává vzhledem k celému objektu, nikoli vzhledem k jedné jeho pozici.

Buffer má jednodušší strukturu, která není vztažena ke konkrétním pozicím, které je nutno postupně obsazovat jako v předešlém případě. To znamená, že po uplynutí nastaveného procesního času je MU odeslán bez nutnosti obsazení všech pozic. *Buffer* může být nastaven jako typ first in first out nebo jako last in first out.

4.3.1.10 DismatleStation

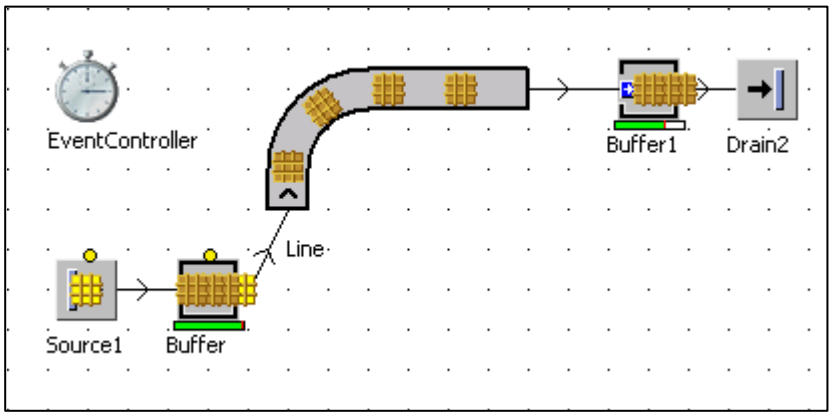
Plní funkci opačnou k *AssemblyStation* a dokáže tak přicházející materiál rozdělit, nebo rozebrat, na více jednotlivých částí. Využití je například u paletového balení, kdy chceme paletu oddělit od výrobků a ty chceme navíc následně odebírat postupně. *DismatleStation* může buď zrušit vzájemné přiřazení objektů, které vytvořila *AssemblyStation*, nebo může z jednoho příchozího MU vytvořit více nových.

4.3.1.11 Store

Prezentuje skladovací prostory, které jsou organizovány ve tvaru matice volných pozic. Dokud je některá z pozic volná, je možné ji obsadit a jsou tedy přijímány další MU. Jedná se o pasivní statický objekt, není zde žádný procesní čas ani výstupní strategie. Jedinou možností, jak sklad vyprázdnit, je využití externího řízení, konkrétně metody.

4.3.1.12 Line

Objekt *Line* využijeme při modelování běžných typů dopravníků. Plant Simulation k tomu nabízí možnost modelovat dopravník graficky tak, aby svými rozměry a tvary odpovídal reálnému zařízení. *Line* přesouvá materiál určitou konstantní rychlostí, jeho kapacita je buď omezena na konkrétní číslo výrobků, nebo je vypočítána na základě délky linky a rozměrů jednotlivých MU. Vhodným nastavením akumulace lze modelovat jak pásový dopravník, kdy se po zablokování výstupu objekty na konci sjíždí, tak i například řetězový dopravník, u kterého se při blokaci objekty nesjíždí a linka se pouze zastaví.



Obr. 4.11 Modelový příklad s využitím dopravníku

4.3.1.13 AngularConverter, Turntable a Turnplate

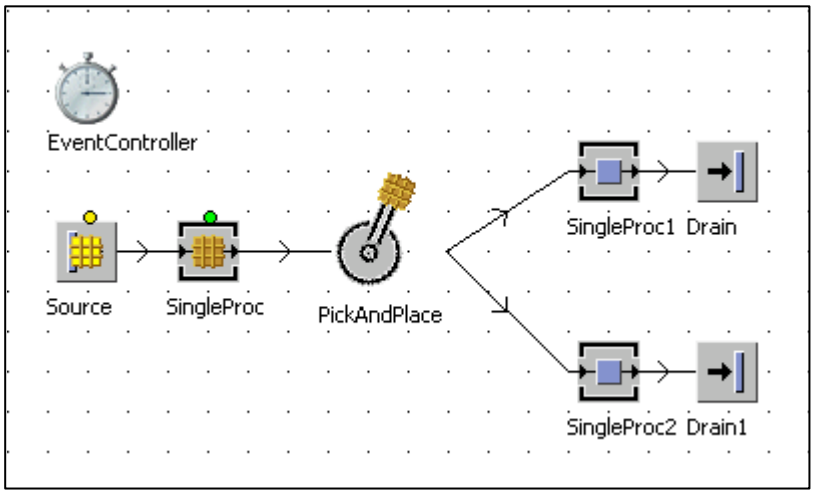
Protože předchozí *Line* nenabízí tolik možností pro manipulaci s materiálem, existují zde další objekty, které se na tyto operace specializují. *Turntable* je otočný kus linky, který

umožňuje otáčení se souběžným posunem MU. Použijeme ho jak pro samotné otočení materiálu, tak například jako dopravní uzel, pokud potřebujeme linku rozvětvit na více částí. Stejnou funkci, avšak bez současného posunu, plní *Turnplate*.

AngularConverter je rohový kus linky, který zachovává orientaci MU. Princip je takový, že dopravovaný materiál dojede na konec jedné části, tam se zastaví a následně se rozjede kolmým směrem vůči původnímu směru pohybu.

4.3.1.14 PickAndPlace

Znázorňuje robotickou operaci pick and place, tedy přenos MU z jednoho místa na druhé. Můžeme přenášet i více dílů najednou a na více různých pozic. Pozice jsou definovány pomocí tabulky úhlů, do kterých je robota třeba natočit. Tabulka se při propojování objektů vytváří automaticky, ale je možno ji napsat či upravovat i ručně. Kromě úhlů zadáváme i časy přesunů mezi konkrétními pozicemi.



Obr. 4.12 Robotická operace pick and place

4.3.1.15 Track

Modeluje cestu pro transportéry - transportér, který patří mezi MU, je jediný, který může tuto trasu využít. Stejně jako u dopravníku, i zde se může pohybovat takový počet

transportérů, kolik se jich se svými rozměry na trasu vejde, popřípadě můžeme maximální kapacitu pevně stanovit. *Track* může být jednosměrná i obousměrná.

4.3.1.16 Sorter

Sorter, na rozdíl od objektu *Buffer*, umožňuje změnu pořadí MU, které jsou v něm momentálně uloženy. Kapacita, kritérium třídění, doba třídění a způsob třídění jsou nastavitelné ve vlastnostech objektu. Pro složitější kritéria je možno přidat externí řízení za pomoci metod a programu v jazyku SimTalk.

4.3.1.17 FlowControl

Není objektem, který modeluje reálně vykonávaný proces na MU. Mluvíme spíše o organizačním prvku v simulaci. *FlowControl* se stará o směrování toku materiálu mezi dvěma a více různými statickými objekty. Na základě stanoveného kritéria je materiál odeslán po větvi ke správnému následníkovi. Kromě výstupní strategie můžeme nastavit i strategii vstupní.

4.3.2 Resource objects

Zdroji jsou v Plant Simulationu myšleny zaměstnanci, tedy pracovníci obsluhující pracovní stanice. Zde pak fungují jako operátoři stroje, jako technici pro údržbu nebo přenáší díly od jedné stanice ke druhé.

4.3.2.1 Workplace

Definuje místo pro vykonávání práce, přiřazuje se k pracovní stanici (nejčastěji *SingleProc*), u které člověk pracuje. Pracovištěm tak spojujeme určitou událost stanice a požadavek na obsluhu (například oprava). *Workplace* může obsadit pouze jeden pracovník.

4.3.2.2 FootPath

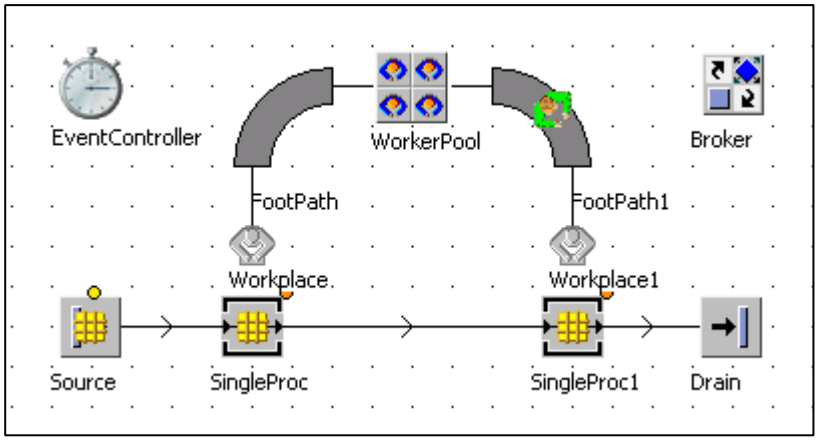
Cesta, po které se pracovníci pohybují. Měla by vždy existovat alespoň jedna cesta z *WorkPool* ke každému pracovišti. Pokud cest existuje více, pracovník při přemisťování zvolí kratší z nich.

4.3.2.3 WorkPool

Jedná se o místo, kde jsou „shromažďováni" právě nevyužití pracovníci. Ti jsou zde zároveň vytvářeni na základě Creation Table, kde je definován jejich počet a práce, kterou vykonávají. V případě potřeby jsou vysláni na konkrétní pracoviště.

4.3.2.4 Worker

Samotný pracovník. Lze mu nastavit několik základních vlastností, jako je jeho rychlost chůze, efektivita, vykonávaná práce nebo počet dílů, které je schopen nést najednou.



Obr. 4.13 Příklad stanic, které jsou obsluhovány pracovníky

4.3.2.5 Broker

Zajišťuje vypravení pracovníka ze stanoviště *WorkPool* ke stanici, která poslala požadavek na obsluhu. Tímto způsobem řídí pohyb pracovníků, nepotřebuje žádné další nastavení. Pracovník zůstává u stanice po celou dobu výkonu práce.

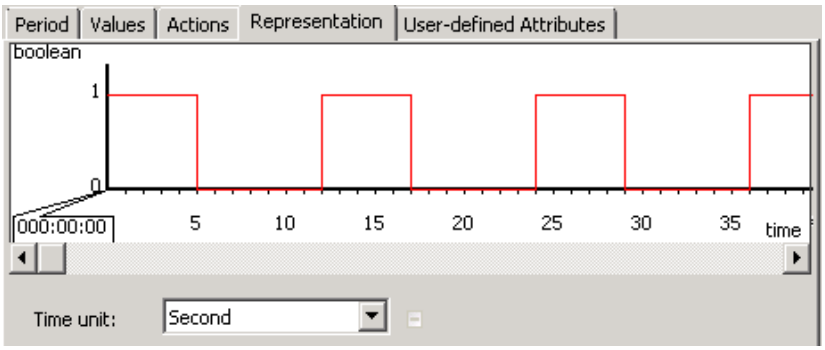
4.3.2.6 ShiftCalendar

Umožňuje nastavení směn, přestávek, svátků, a podobně. Jedná se tedy o časový rozvrh pracovníků, ale i ostatních zařízení, která se na základě aktuálního času můžou například vypínat.

4.3.3 Information flow

Do této skupiny patří objekty, které pracují s informacemi, daty a atributy simulace. Jsou zde objekty pro export a import dat, které mohou tvořit interface mezi Plant Simulationem a dalšími programy. Dále zde máme možnost vytvářet různé druhy seznamů, zásobníků a tabulek. Konkrétně se jedná o *CardFile, QueueFile, StackFile, TableFile* a *TimeSequence*. S těmi buď následně přímo spolupracují další objekty z material flow (zadáme pouze cestu k objektu) nebo k nim přistupujeme přes vlastní metody. Možná nejdůležitější objekt pro vlastní řízení se nazývá *Method*. Jde vlastně o rozhraní mezi grafickým modelováním a programovým prostředím. Pro programování se používá jazyk SimTalk vyvinutý právě pro tyto účely. Problematice programování v Plant Simulationu je věnována jedna z následujících kapitol. S *Method* úzce spolupracuje *Variable*, kde jsou definovány globální proměnné pro přístup napříč různými objekty.

Do poslední podskupiny můžeme zařadit *Trigger* a *Generator*, které se starají o řízené generování MU a o spouštění metod v předem stanovených intervalech. V principu mluvíme o objektech zajišťujících synchronizaci určitých částí linky.



Obr. 4.14 Objekt Trigger s vyobrazeným časovým průběhem

4.3.4 User interface/statistics objects

Soubor objektů pro interakci s uživatelem lze nalézt v Toolboxu pod záložkou User Interface. I když je tato záložka defaultně neobsahuje, jsou do této kapitoly zahrnuty i další statisticko-analytické objekty z Tools, které sem řadíme také. Mezi základními prvky pro interakci s člověkem, rozumějme operátorem, najdeme i grafické prvky pro ovládání, jako jsou tlačítka, checkboxy a dropdownlisty.

4.3.4.1 Display

Informuje uživatele o proměnných veličinách v simulaci. Těmi mohou být atributy nebo vlastní proměnné. Hodnota je zobrazována jako číslo (případně text) s možností vykreslení v jednoduchém bargrafu, který je v rozmezí přednastavené minimální a maximální hodnoty. Aktualizace veličin probíhá automaticky po určitém čase nebo vždy po změně o určitou velikost.

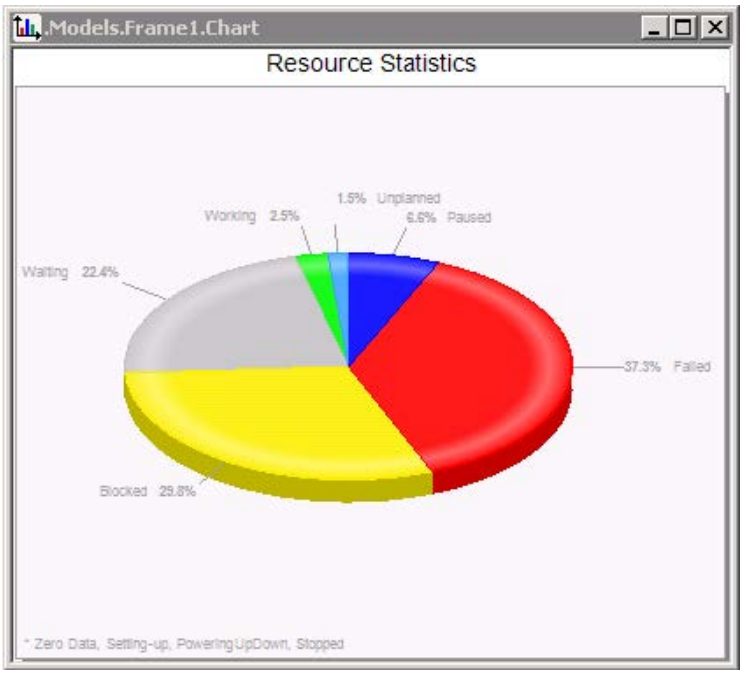
4.3.4.2 Dialog

Umožňuje tvorbu vlastních dialogových oken, například pro nově vytvořené objekty. U standardních objektů slouží dialogové okno pro nastavení některých základních atributů a pro zobrazení statistických údajů dané výrobní jednotky. Dialog je dále vhodné pro přehlednost použít i v souvislosti se samotným programováním metod, kde může tvořit grafické uživatelské rozhraní, například pro jednodušší zadávání vstupních parametrů.

4.3.4.3 Chart

Objekt *Chart* dokáže vybraná data z Plant Simulationu zobrazit graficky. První možností je přímo zadat vstupní kanál (parametr, atribut), jehož vývoj chceme sledovat. Navíc si lze zvolit, zda budeme průběh sledovat s určitou vzorkovací frekvencí, nebo zdali se spokojíme pouze s vykreslením v případě změny hodnoty. Druhou variantou je pak zobrazení již nahraných dat, nejčastěji uložených v tabulce nebo podobné struktuře. *Chart* obsahuje Statistic Wizard, přes který si lehce naklikáme, co přesně chceme v grafu vidět. Data lze zobrazit různým způsobem, na výběr jsou možnosti chart, histogram

a plotter. Defaultně je zvolen chart, který zobrazuje číselnou hodnotu nebo množinu číselných hodnot, a to v různé formě, například jako sloupcový graf nebo koláčový diagram. Histogram ukazuje četnost hodnot pro jeden nebo více sledovaných kanálů. Poslední možnou volbou je plotter, který dokáže vykreslovat průběh veličin v čase. Při analýze již nahraných dat je k dispozici ještě čtvrtá možnost, a to XY graph, který zobrazuje hodnoty v rovinném souřadnicovém systému. Kromě výše uvedeného zde existuje množství dalších parametrů pro uživatelské přizpůsobení výsledného grafu.



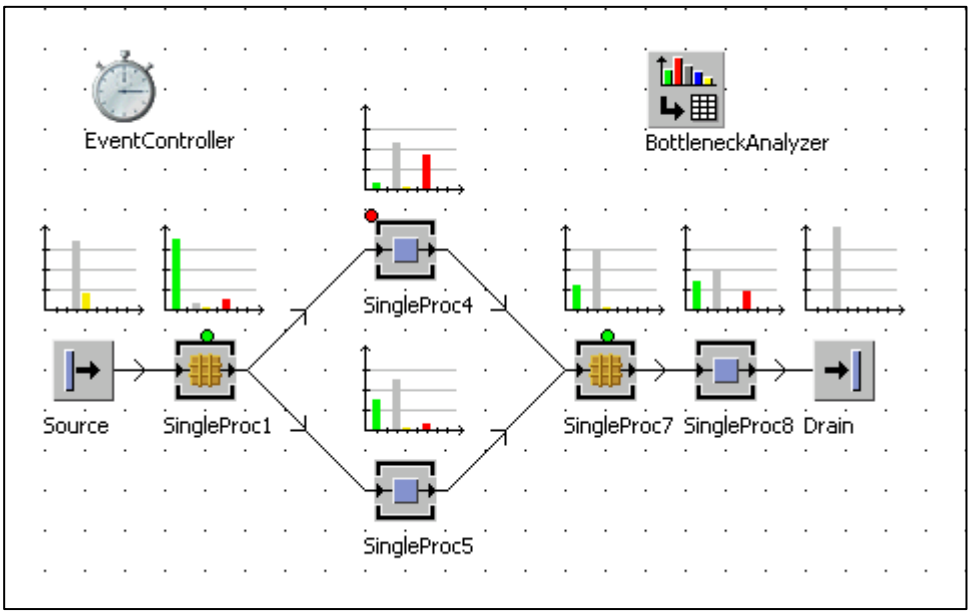
Obr. 4.15 Koláčový diagram vygenerovaný objektem Chart

4.3.4.4 Report

Celá simulace pracuje s velkým množstvím dat a informacemi, které jsou potřeba nějakým přehledným způsobem zobrazit. K tomu slouží report, který vybraná data zobrazí v okně jako HTML soubor. Ten je možno jednoduše vytisknout nebo exportovat. Kromě jednotlivých údajů, hodnot, atributů a tabulek lze do reportu vložit i celé grafy a vytvořené simulační modely. V rámci jedné sestavy můžeme vytvořit několik stránek, jejichž vzhled lze individualizovat.

4.3.4.5 BottleneckAnalyzer

Pro prvotní přehled o základních defaultních statistikách se nabízí využití *Bottleneck* analyzátoru. Jeho použití je jednoduché, stačí si zvolit, zda chceme sledovat objekty pro zpracování výrobků, jejich transport nebo ukládání, popřípadě všechny najednou, a spustit analýzu. Nad objekty se zobrazí malý graf, který informuje o vybraných statistikách. Výstupní data analýzy je možno zobrazit i v tabulce seřazené podle stanoveného kritéria.

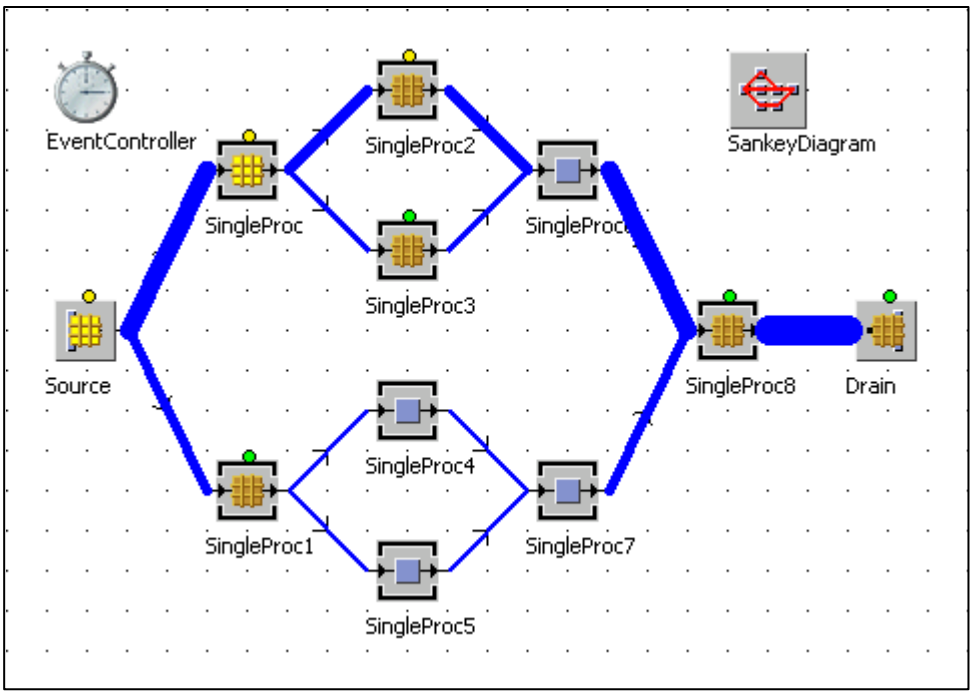


Obr. 4.16 Výsledek zobrazený Bottleneck analyzátorem

4.3.4.6 SankeyDiagram

Pro grafické znázornění toku materiálu používáme Sankeyův diagram. Ten zobrazuje počet mobilních objektů průchozích danou cestou vzhledem k celkovému počtu vygenerovaných objektů. Grafický výstup je v podobě linek mezi objekty, kdy šířka linky je přímo úměrná počtu průchozích MU. Konkrétní podoba je vidět na modelovém příkladu na obr. 4.17. Jedná se o jednoduchý statistický nástroj, jeho jediným

nastavitelným parametrem je, kromě grafiky samotných linek, typ mobilního objektu nebo objektů, které chceme sledovat.

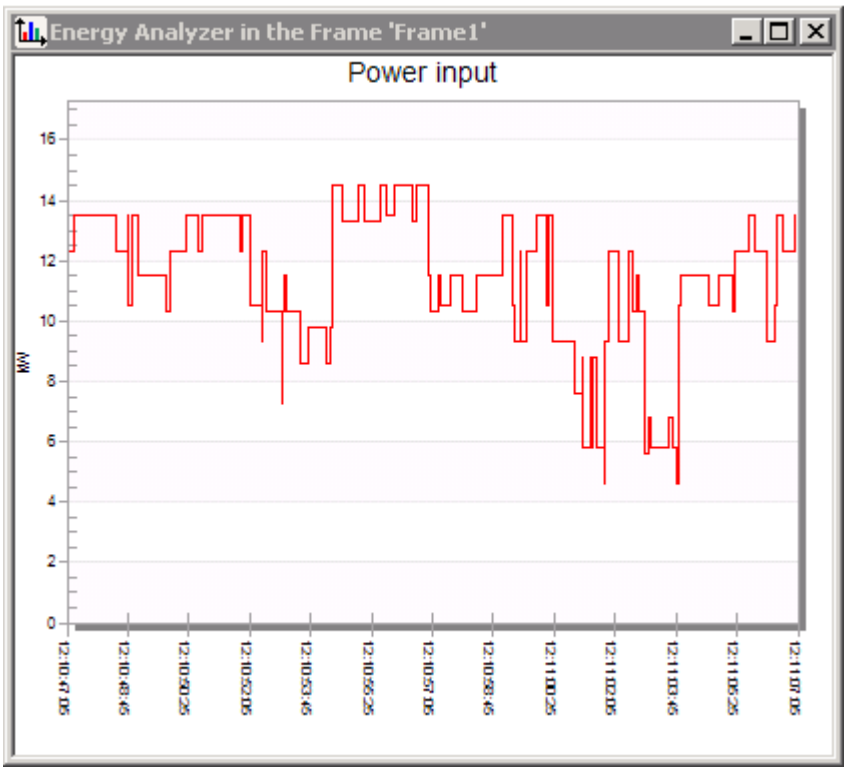


Obr. 4.17 Sankeyův diagram pro zobrazení toku materiálu

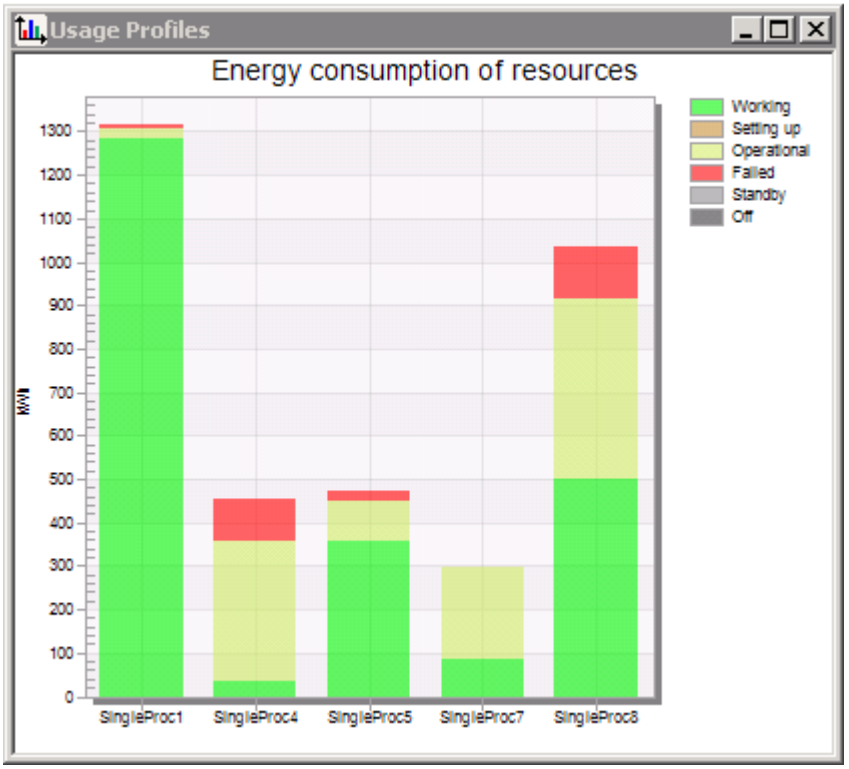
4.3.4.7 EnergyAnalyzer

Verze Plant Simulation 11 umožňuje, na rozdíl od předchozích, měření energetické spotřeby objektů za pomoci speciálního nástroje. Objekty, které měřením spotřeby disponují, mají předdefinované atributy pro zadání jejich příkonu v různých provozních režimech, ve kterých se dané zařízení může ocitnout. Konkrétně jde o stavy Operational, Working, Setting-up, Failed, Standby a Off. Ne všechny objekty nabízejí veškeré tyto módy. Kromě příkonu jsou zde nastavitelné časy přechodů mezi jednotlivými stavy.

Po vložení *EnergyAnalyzeru* stanovíme, u jakých objektů chceme spotřebu energie sledovat. Výsledky jsou zobrazeny graficky a hodnoty se zároveň vygenerují i do tabulky. Vykreslit lze průběh celkové spotřeby linky, podíl jednotlivých energetických stavů objektu na jeho celkové spotřebě a grafickou vizualizaci, která v modelu barevně označí objekty v závislosti na jejich spotřebě.



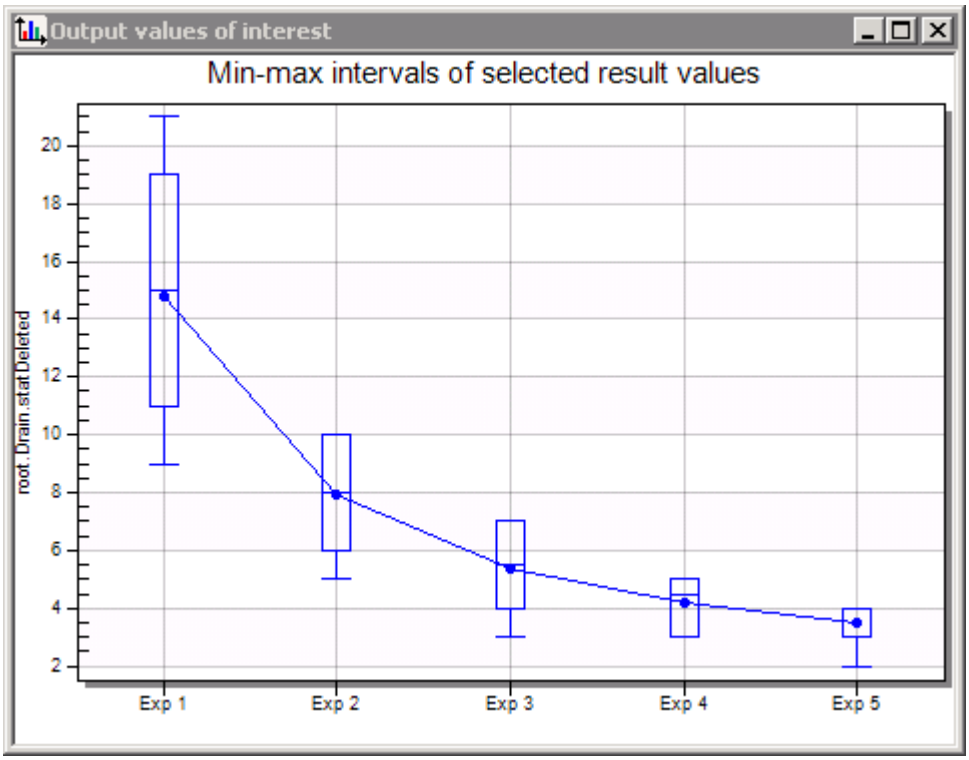
Obr. 4.18 Časový průběh spotřeby celé linky



Obr. 4.19 Spotřeba energie objektů v různých energetických stavech

4.3.4.8 *ExperimentManager*

Po vytvoření simulačního modelu je třeba tento model odsimulovat. Pro nalezení optimálních hodnot parametrů je nutno provést desítky různých experimentů, které nám napoví, jak se změna parametru projeví na námi sledovaném výstupu. *ExperimentManager* slouží právě pro tyto účely. V prvé řadě definujeme výstupní veličiny, které chceme sledovat. Výstupní veličinou může být jakýkoli atribut jakéhokoli objektu nebo metoda. Následně stanovujeme vstupní veličiny, které mají vliv na chování linky, například procesní časy, poruchy a podobně. U těchto vstupních veličin dále nastavujeme buď jejich konkrétní hodnoty, které budeme chtít testovat, nebo rozsah a inkrementační konstantu, se kterou se parametr bude měnit. Důležité je taktéž v *EventControlleru* určit celkový čas, po který se má experiment testovat. Takto lze jednoduchým způsobem vytvořit několik experimentálních simulací, které se postupně automaticky provedou. Po dokončení jsou k nahlédnutí i uložení výsledky a porovnání. Výsledek lze zobrazit v podobě tabulky, reportu nebo grafu.



Obr. 4.20 Grafické zobrazení výsledků experimentu

ExperimentManager je již propracovaný nástroj s rozsáhlými možnostmi nastavení a přizpůsobení konkrétním potřebám. Nutno upozornit, že jeho funkcionalita je závislá na konkrétní licenci Plant Simulationu.

Experiment	root.Drain.statDeleted	Standard Deviation	min	max	Left int. bound	Right int. bound
Exp 1	14,8	4,492	9	21	11,584	18,016
Exp 2	7,9	1,912	5	10	6,531	9,269
Exp 3	5,4	1,43	3	7	4,376	6,424
Exp 4	4,2	0,919	3	5	3,542	4,858
Exp 5	3,5	0,707	2	4	2,994	4,006

Tab. 4.1 Přehled výstupu modelového experimentu

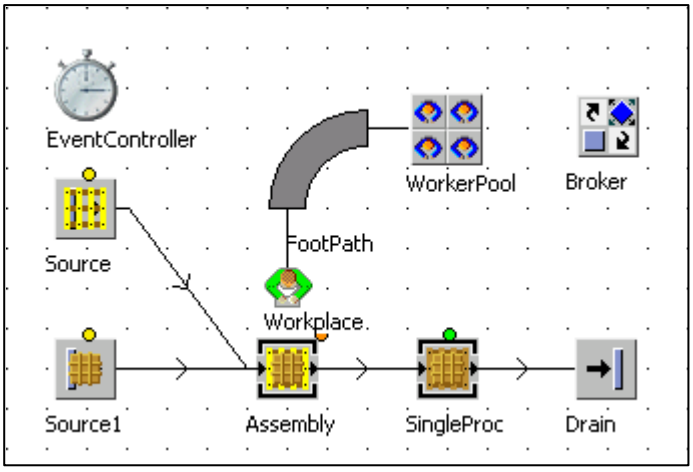
4.3.4.9 GeneticAlgorithms Wizard

Pro pokročilejší metody hledání optimálních hodnot atributů a pro celkovou optimalizaci modelovaného výrobního procesu Plant Simulation nabízí řešení v podobě genetických algoritmů. Genetické algoritmy jsou stochastické optimalizační postupy, které ve většině případů poskytují pouze přibližné řešení, nicméně pro běžné využití dostačující. V principu zde postupujeme stejně jako v případě *ExperimentManageru*, nejprve zadáme parametry, které chceme optimalizovat vzhledem k parametrům, jejichž výstupní chování sledujeme. Změna hodnot atributů však není pevně stanovená, ale je postupně upravována právě na základě probíhajících simulací. Kvalita návrhu řešení se hodnotí podle hodnoty fitness funkce, kterou je potřeba při návrhu stanovit. Dále určujeme počet a velikost generací, tedy hloubku, do jaké bude genetický algoritmus pracovat. Jedná se již o komplexní nástroj s rozsáhlými možnostmi nastavení. Tyto možnosti i samotné použití *GAWizardu* jsou opět závislé na licenci programu.

4.3.5 Mobile objects

O mobilních objektech (MU) jsme se v předešlých kapitolách zmínili několikrát. Jak již bylo uvedeno, tvoří spolu se stacionárními objekty celou kostru simulačního modelu. Zpočátku jsou vytvořeny, postupně prochází skrz model, a na konci opět zanikají. To, jakým způsobem virtuálním modelem projdou je vlastně podstatou a smyslem celé simulace.

MU představují zpracovávané výrobky, jejich součásti, polotovary, díly, konečné produkty, ale i pohybující se transportéry nebo přepravní materiál. Každý tento objekt má definované svoje základní vlastnosti, jako je rozměr, hmotnost, dále třeba barvu nebo i cenu. V Plant Simulationu mají pro přehlednost i své symbolické grafické znázornění, aby bylo vidět, kde přesně a v jakém stavu se nacházejí. Můžeme využít již nadefinovaných objektů nebo si vytvořit svoje vlastní. Mezi základní mobilní objekty patří *Entity*, *Container* a *Transporter*. Jak názvy napovídají, *Entity* je základní materiálový prvek, který je na lince zpracováván. *Container* slouží k přepravě a můžeme si ho představit jako paletu, krabici nebo přepravní kontejner. *Transporter* se zde prezentuje jako jakýkoli druh nezávisle pohybujícího se přepravníku - tedy například vysokozdvižný vozík nebo nákladní automobil. Slovo nezávisle zde musíme brát s rezervou, protože i transportér se musí pohybovat pouze po stanovených cestách, nicméně je potřeba ho odlišit od stacionárních dopravníků.

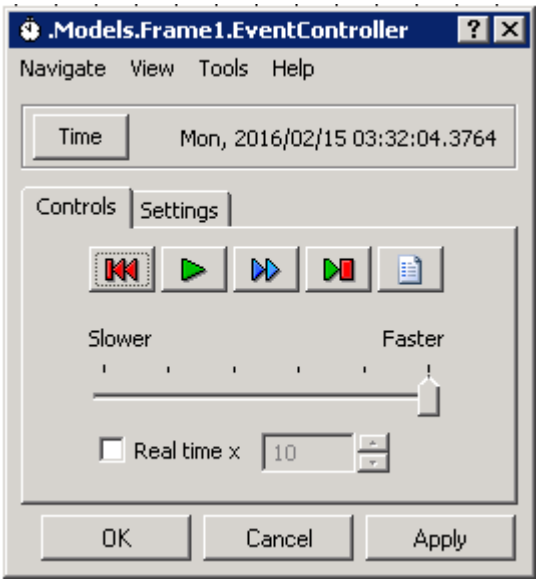


Obr. 4.21 Příklad skládání MU typu Entity na MU typu Container

4.3.6 General objects

Kromě již popsaných objektů jsou dále zapotřebí ještě další prvky, bez kterých se při modelování neobejdeme. V prvé řadě jde o *Connector*, který slouží k propojení objektů a stanovuje tak vzájemnou návaznost. Další důležitou součástí je *EventController*, který řídí jednotlivé události a celý proces simulace z hlediska času. Základním parametrem je čas, po který má být simulace testována. Ten může být stanoven absolutně, tedy od přesného data a času až opět po konkrétní datum a čas, nebo relativně, kdy čas běží od nuly. Průběh lze sledovat v reálném čase, ale je samozřejmě na místě rychlost zvýšit a odsimulovat vše co nejrychleji.

Někdy, například při vytváření rozsáhlých výrobních zařízení, je potřeba určitou část modelu pro přehlednost zahrnout do jednoho samostatného objektu. Takovýto objekt vytvoříme v novém *Framu*. Pro možnost jeho propojení s dalšími objekty využijeme prvku *Interface*. Takto lze celou výrobu modelovat za pomoci jen několika buněk, kdy každá buňka může obsahovat jakkoli složitou strukturu.



Obr. 4.22 EventController

4.4 SimTalk

Základní chování Plant Simulationu je v mnoha případech nedostačující. Předdefinované objekty a jejich vzájemné provázání nemusí věrohodně napodobovat reálný systém. Stejně tak samotný proces simulace je často nutné „ušít" na míru. Pro tyto případy program nabízí prostor pro nadefinování vlastních objektů, expertíz, struktur, vlastností linky a jejího chování. V krajním případě je možné celý simulační proces od samotného návrhu až po statistické expertízy a optimalizace ručně naprogramovat. Děje se tak za pomoci jazyka SimTalk, který byl vytvořen právě pro tyto účely. Díky svému specifickému zaměření obsahuje metody a funkce pro snadnou práci s objekty. Mezi standardními datovými typy nalezneme kromě těch základních například acceleration, speed, time, datetime a jiné.

4.4.1 Method

Pro provázání objektů a námi napsaného kódu slouží metody. *Method* je sám o sobě objektem, po jeho rozkliknutí se však místo dialogového okna zobrazí okno textové, do kterého zadáváme vlastní kód. Metoda má danou strukturu, která je vyobrazena na obr. 4.23. V první řadě se definují vstupní parametry, které mohou být předávány při volání metody, a v těsné návaznosti i parametry výstupní. Dále deklarujeme lokální proměnné a nakonec i samotný kód.

Pro vykonání naprogramované posloupnosti je nutno metodu zavolat. Volání je způsobeno určitou akcí, například stisknutím tlačítka, vstupem MU do objektu nebo jednoduchým triggerováním za pomoci prvku *Trigger*.

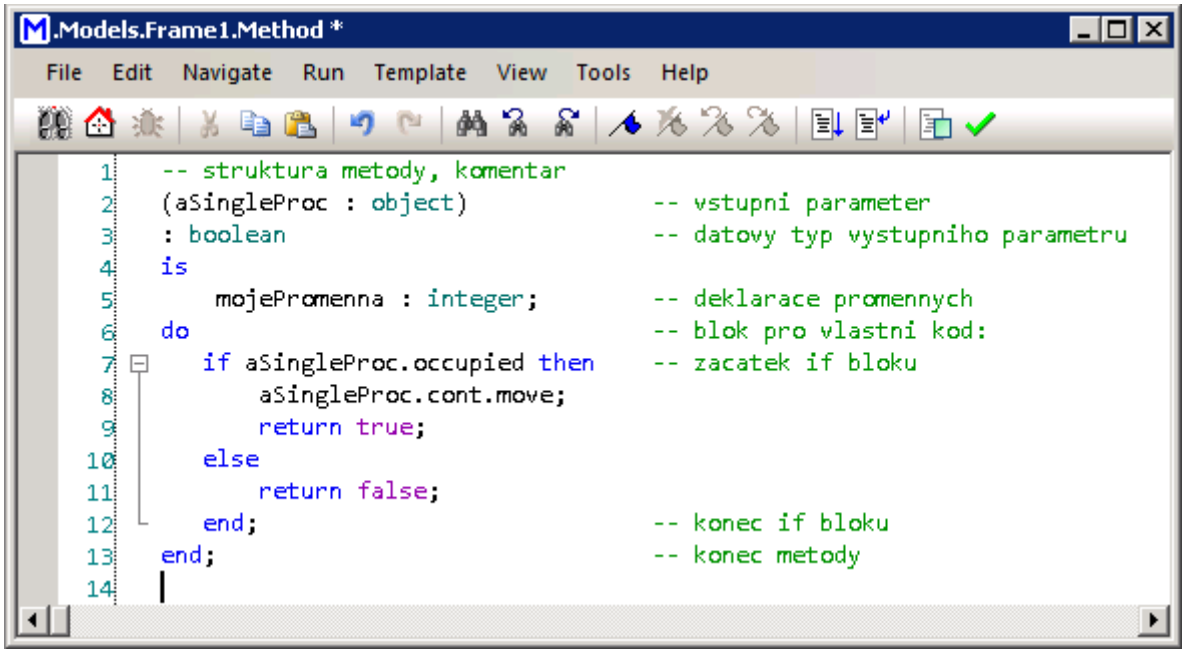
4.4.2 Proměnné

Při deklaraci proměnné alokujeme určité místo v paměti, kam následně bude ukládána její hodnota. SimTalk pracuje s lokálními proměnnými, ke kterým lze přistupovat pouze v rámci metody, a globálními proměnnými. Globální proměnnou opět vkládáme do

projektu jako samostatný objekt, konkrétně jde o *Variable*. *Variable* je dosažitelná z každé metody i z každého *Framu*.

4.4.3 Debugger

Součástí editoru pro psaní kódu je i debugger. Ten využijeme jak při kontrole zdrojového kódu, tak i při jeho testování. Metodu je možno procházet krok po kroku, zjistíme tak přesně, kde se co děje. Pokud nastane chyba během simulace, debugger automaticky zobrazí chybové hlášení.



Obr. 4.23 Obecná struktura metody

5 MODELOVÁNÍ VÝROBNÍ LINKY

Tato část se zabývá praktickým příkladem využití programu Plant Simulation pro simulaci konkrétního výrobního pracoviště.

5.1 Zadání

Úkolem je vytvoření virtuálního modelu dodané výrobní linky v Plant Simulationu. Ta je ve formě eBOP (Electronic Bill of Process) projektu pro Process Designer a Process Simulate. Díky tomu máme k dispozici jak přesné prostorové rozmístění pracovišť, tak i Ganttův diagram všech probíhajících procesů. Samotná simulace operací v rámci Process Simulate není součástí dodaného projektu.

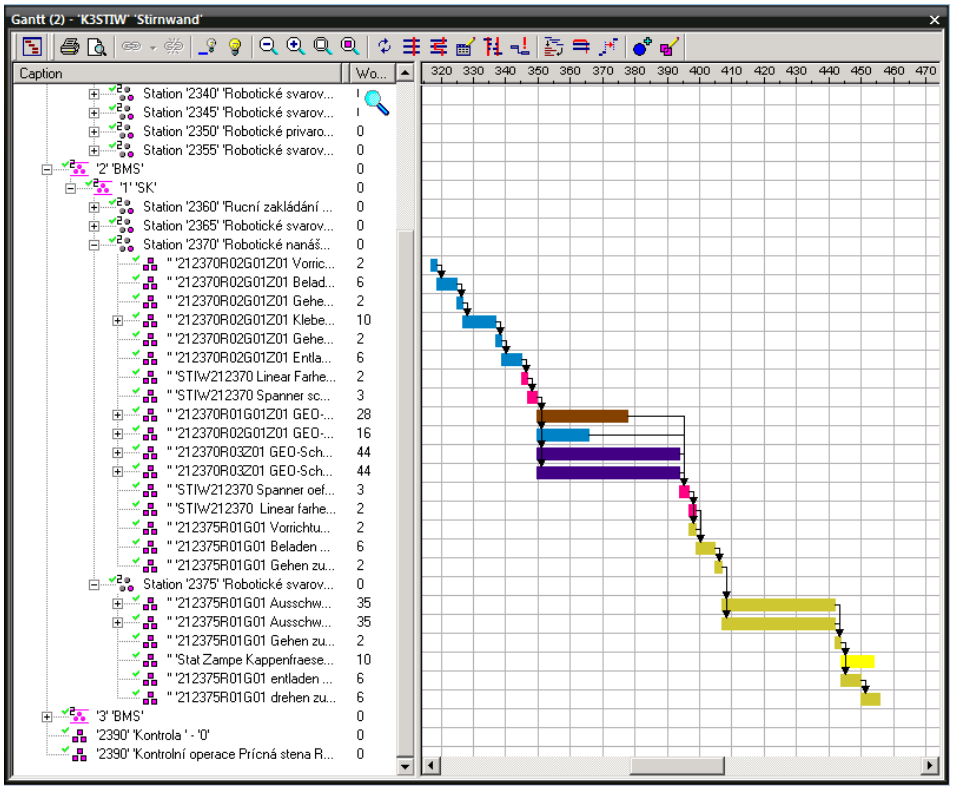
Výsledný model bude následně podroben analýze, která nám poskytne základní informace o produkci a efektivitě linky. Na základě toho bude provedeno několik experimentů, jejichž úkolem bude „optimalizovat" linku vzhledem k předem stanoveným kritériím. Soustředit se budeme především na energetickou úsporu, kde budeme brát v potaz možnost přepínání robotů do energeticky úsporných stavů. Cílem je dosáhnout menší celkové spotřeby energie s minimálním dopadem na produkci. Kromě energetické analýzy budeme zkoumat vliv náhodných poruch zařízení na celkový chod linky a různé možnosti pozastavování výroby v závislosti na pracovních směnách a během pauz.

5.2 Popis linky

Dodaná výrobní linka je vlastně předpřipraveným modelem pro zhotovení její reálné obdoby. Protože dodaný projekt neobsahuje samotnou simulaci, jsou konkrétní operace a procesy doladěni až u fyzické linky. To může znamenat, že si reálná a virtuální linka, ze které vycházíme, nemusí ve všech parametrech odpovídat. Náš model v Plant Simulationu vychází právě z modelu virtuální linky v Process Designeru, protože data a parametry fyzické linky nebyly k dispozici. Z tohoto důvodu zde v porovnání s reálnou linkou může docházet k lehkým nepřesnostem.

Linka je součástí svařovny v továrně Škoda Auto a má za úkol sestavování dílu nazývaného příčná stěna (v terminologii VW koncernu Stirnwand) pro model Octavia. Jedná se o část, která se nachází mezi kabinou řidiče a motorovým prostorem, pod palubní deskou. Celé pracoviště se dělí na další 3 podpracoviště označené jako 1 BMS, 2 BMS a 3 BMS. 1 BMS a 2 BMS se dělí na další 2 buňky pojmenované 1 SK a 2 SK, 3 BMS obsahuje pouze jednu buňku 1 SK. V každé buňce je pak několik stanic, ve kterých jsou popsány robotické a ostatní operace s materiálem.

Linka pracuje se dvěma variantami dílů. Podle toho, který díl se zrovna zpracovává, se volí předdefinované operace s různými procesními časy. Pro zjednodušení modelu předpokládáme pouze jednu variantu. Přehled probíhajících procesů podle stanovišť je obsahem přílohy 9.1. Ukázka Ganttova diagramu, z kterého vycházíme, vygenerovaná programem Process Designer, je na obr. 5.1.



Obr. 5.1 Část Ganttova diagramu

5.3 Definice vlastních objektů

Ještě před samotným vytvářením struktury modelované linky v Plant Simulationu je třeba přizpůsobit si některé objekty našim potřebám. Tyto námi vytvořené objekty následně využijeme při modelování.

Protože se budeme věnovat především energetickým experimentům, v základu nabízené objekty vhodně upravíme tak, aby umožňovaly řízené přepínání energetických módů. Již defaultní objekty těmito stavy disponují, nicméně jde pouze o jednoduchou funkcionalitu, kterou je nutno rozšířit. Kromě samotného řízení rozšíříme objekty i po vizuální stránce, aby bylo na první pohled patrné, v jakém stavu se momentálně nacházejí a jakou operaci vykonávají. Před popisem vlastních objektů stručně popíšeme standard Profienergy, který s řízením energetických stavů průmyslových zařízení úzce souvisí.

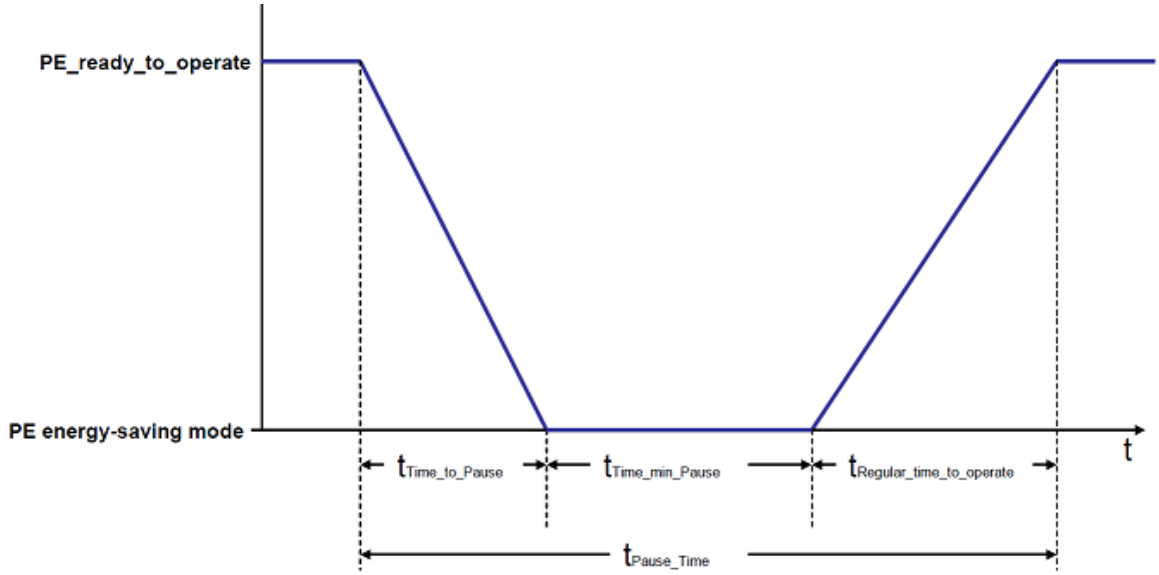
5.3.1 Profienergy

Mnohá moderní průmyslová zařízení dnes podporují standard Profienergy. Tento standard stanovuje několik energetických profilů, ve kterých se dané zařízení může nacházet. To obecně obsahuje několik dílčích periferií a podsystémů, a každý energetický profil definuje, které části mají běžet a které se naopak odpojí. Tím je docíleno energetické úspory. Samozřejmě, ne v každém stavu je zařízení plně funkční. Součástí Profienergy je navíc sada příkazů, s pomocí kterých se kompatibilní zařízení ovládají.

Name	Drive Bus OFF	Hibernate	Ready to Operate
PE Mode ID	0x01	0xFE	0xFF
Min Pause Time	25s	1m 50s	0
Time to Pause	5s	50s	0
Time to Operate	20s	50s	0
Time min Pause (length of stay)	0	10s	0
Time max Pause (length of stay)	-1ms	-1ms	-1ms
Mode Power Consumption	0,15 kW	0,03 kW	0,22 kW
Energy Consumption to pause	0	0	0
Energy Consumtion to operate	0	0	0

Tab. 5.1 Specifikace energetických stavů KUKA

Naše vzorová výrobní linka obsahuje roboty od firmy KUKA. Důležitější je pro nás v tomto momentě robotický kontrolér, který robota ovládá, právě ten má nadefinovány energetické profily podle Profienergy. Jedná se o profily Ready_to_Operate, Hibernate a Drive_Bus_OFF. Každý stav má svá specifika, nejdůležitější pro nás jsou shrnuty v tabulce 5.1. Zajímá nás především příkon zařízení v konkrétním stavu a časy přechodů mezi stavy. Co který čas vyjadřuje, nám ukazuje graf na obr. 5.2.



Obr. 5.2 Přechod zařízení (robota) do úsporného módu

Stav PROFIenergy	Stav Plant Simulation
Ready to Operate	Operational
Drive Bus OFF	Standby
Hibernate	Off

Tab. 5.2 Přiřazení ke stavům v Plant Simulation

Energetické profily v Plant Simulationu mají trochu odlišný charakter než ty podle standardu Profienergy. Defaultní objekty, například *SingleProc* nebo *PickAndPlace*, umožňují změnu standardního operačního režimu Operational na Standby nebo Off. V první řadě je tedy nutné spárovat tyto profily s profily, které máme v robotickém

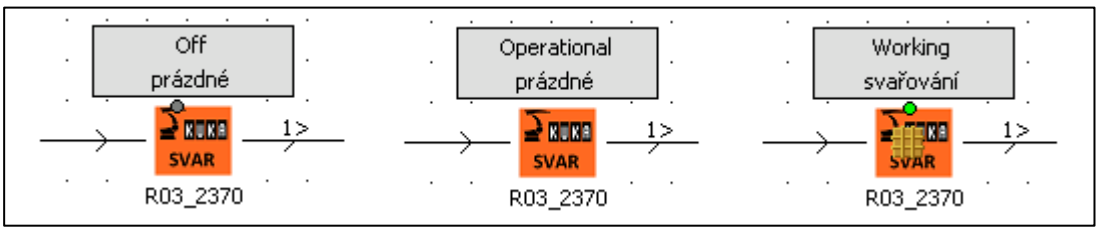
kontroléru KUKA. Přiřazení vidíme v tabulce 5.2. Stav Operational představuje robota, který je spuštěný, ale momentálně nevykonává žádnou činnost. Standby bude simulovat vypínání sběrnice, tento stav však využívat nebudeme. A poslední Off symbolizuje zařízení ve stavu hibernace.

5.3.2 Objekt SV (svařování)

Jedná se o objekt, který modeluje robotickou operaci svařování. Třída je odvozena od klasické operace *SingleProc*, jde tedy o její duplikát, kdy se zkopírují veškeré atributy, nicméně zde neexistuje žádná další provázanost přes dědičnost (viz. kap. 4.2.3). Protože operace svařování je prováděna vždy stejným typem robota, můžeme již v rámci celé této třídy přednastavit atributy, které robot má. Každá instance je po vložení do modelu ihned zdědí, což nám ušetří práci a celý model se tím sjednotí. Nastavíme tedy hodnoty spotřeby energií, poruchy a veškeré další obecné vlastnosti. K dosazení konkrétních hodnot se dostaneme v části věnované experimentům a analýzám. Kromě již uvedeného, přiřazujeme objektu několik vlastních atributů. Jejich výčet a smysl je popsán v tabulce tab. 5.3.

Atribut	Datový typ	Popis
casSpanek	datetime	Ukládá čas, po který je objekt v nečinnosti a bylo by ho tak možno uspat.
druhOperace	string	Popisuje operaci, kterou objekt právě vykonává.
enStav	integer	Ukládá energetický stav jako číselnou hodnotu pro vykreslení grafu.

Tab. 5.3 Vlastní atributy objektů

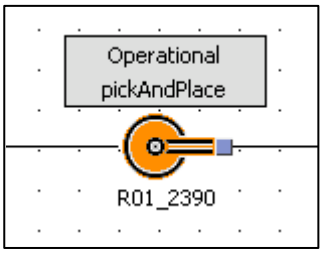


Obr. 5.3 Objekt SV se zobrazením několika stavů

Poslední sadou úprav jsou již úpravy vizuální. Jde o změnu ikony tak, aby náš nově vytvořený objekt byl na první pohled rozeznatelný od ostatních. A dále přidáváme funkci panelu, který textově indikuje aktuální energetický stav objektu - tedy svařovacího robota. Panel zároveň informuje o tom, jakou operaci momentálně vykonává nebo jestli v daný okamžik nepracuje s žádným dílem. Možná zobrazení jsou znázorněna na obr. 5.3.

5.3.3 Objekt PAP (pick and place)

Tento objekt nahrazuje klasickou operaci pickAndPlace operací s rozšířenou energetickou funkcionalitou, podobnou jako v případě svařování objektu SV. Stejně jako v předchozím případě, byl vytvořen duplikací defaultního objektu *SingleProc*. Z praktických důvodů vůbec nevyužíváme předdefinovaného objektu *PickAndPlace*. Děje se tak především z důvodu možnosti modelování robota, který postupně vykonává více různých operací. Tato problematika je podrobně popsána v kapitole 5.3.7. Navíc, protože bereme operaci přesunu materiálu jako jednu událost o konkrétním čase a určité spotřebě, modelováním přes *SingleProc*, namísto standardního *PickAndPlace*, nepřicházíme o žádnou informaci. Energetické spotřeby za různých stavů je opět možno předvyplnit. Vlastní parametry jsou taktéž stejné jako u minulého objektu, stejně tak grafická vizualizace energetického a operačního stavu, ve kterém se robot momentálně nachází. Ikona je i v tomto případě pro přehlednost pozměněna.

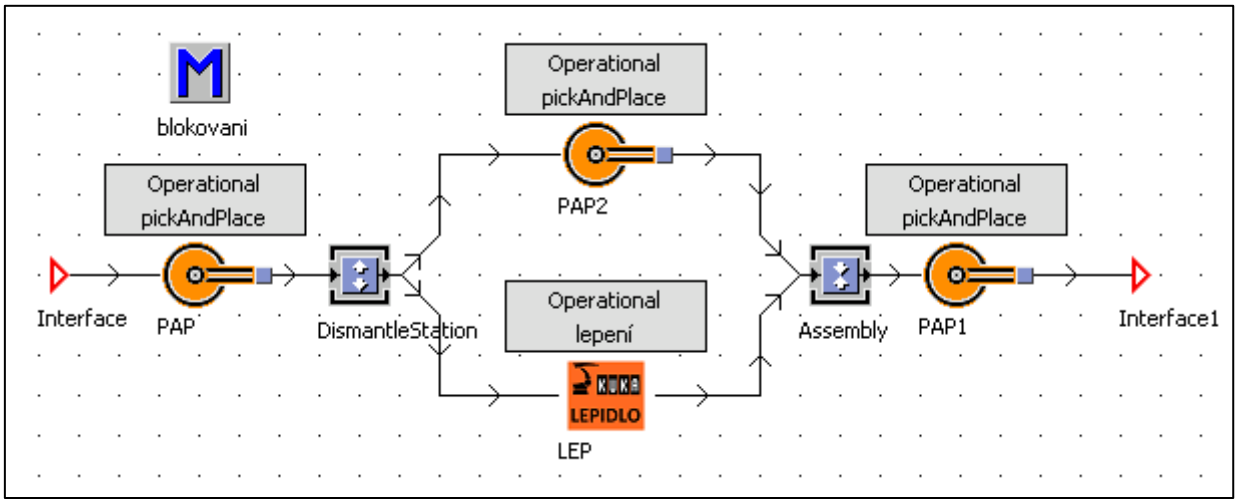


Obr. 5.4 Objekt PAP

5.3.4 Objekt ST_LEP (stacionární nanášení lepidla)

Nanášení lepidla je na rozdíl od svařování kontinuální a na modelování složitější proces. To především z toho důvodu, že se fyzicky jedná o dva spolupracující objekty - robota pohybujícího dílem a stacionárně umístěné „nanášečky“ lepidla poblíž robota. Původní myšlenkou bylo modelovat tento objekt jako spojení několika operací, a to přes *PickAndPlace* a *SingleProc*. Robot nejdříve přemísťuje díl ke zdroji lepidla, dále se spustí samotné nanášení lepidla za současného pohybu dílem, a po dokončení této operace je díl robotem opět přesouván, často již na následující stanoviště v pořadí. Rozdělením na několik operací bychom dostali možnost nastavení parametrů (například poruch a energetických spotřeb) zvlášť pro lepící pistoli a zvlášť pro samotného robota. Kromě toho by bylo na první pohled patrné, že součástí této jedné operace je i přemístění dílu z jednoho stanoviště na další, čili jakási rozšířená pick and place operace. Modelování tímto způsobem však přináší mnoho komplikací.

Prvním problémem je paralelní práce dvou objektů na jednom dílu zároveň, kterou Plant Simulation standardně neumožňuje. Relativně jednoduchým řešením je virtuální rozdělení dílu na dva, kde pak objekty pracují každý s jednou částí, po dokončení paralelní operace dojde opět ke spojení dílů. Jak bude vidět později, s tímto problémem se setkáme i při modelování celé linky a díky popsanému postupu se nakonec nejedná o velkou komplikaci.

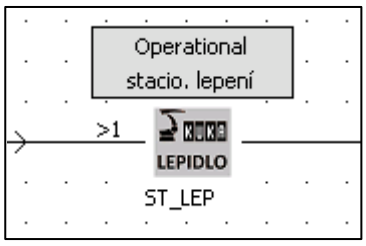


Obr. 5.5 Původní koncept objektu modelujícího nanášení lepidla

Další věcí, kterou je třeba řešit, je určitá synchronizace objektů. Nanášení lepidla musí začít až v momentě, kdy je díl na místě. Realita je taková, že zdroj lepidla je fyzicky oddělené zařízení od samotného robota, nicméně jeho ovládání je řízeno právě robotem podle toho, v jaké části operace se nachází. Kromě toho bychom museli dále, při řešení koncepce uspávání, sjednotit energetické stavy objektů, které ve skutečnosti představují jedno konkrétní zařízení. Tato problematika je detailněji rozebrána v podkapitole 5.3.7 pro robota s více operacemi.

Posledním problémem je samotné použití více objektů pro jednu operaci. Každý objekt totiž dovoluje v jeden okamžik práci s jedním dílem. Pokud bychom modelovali nanášení lepidla za pomoci 4 objektů, jak je uvedeno na obrázku 5.5, bez dalšího softwarového ošetření by se mohlo stát, že v konkrétní moment jednotlivé podoperace pracují s více díly najednou. To však není fyzicky možné. V konečné fázi by tak došlo ke značnému zesložitění celého modelu. Proto bylo rozhodnuto, že celou operaci nanášení lepidla bude představovat pouze jeden upravený objekt *SingleProc*, podobně jako v případě sváření. Přijdeme tak sice o možnost definovat spotřebu energie a poruchovost zvlášť pro robota a pro přístroj nanášející lepidlo, ale mnohonásobně se nám zjednoduší struktura, řízení modelu a jeho následná analýza. Mimo to, spotřebovanou energii a možnost poruchy lze v rámci dvou objektů ještě relativně dobře odhadnout a dopočítat. Pro simulaci máme navíc k dispozici pouze orientační spotřeby jen pro samotné roboty, proto se soustředíme na energetickou úsporu především u nich.

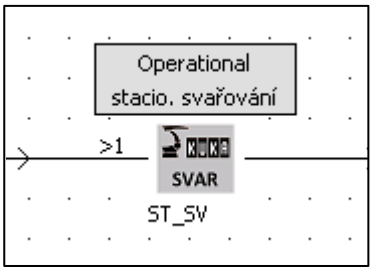
Stejně jako u ostatních námi vytvořených objektů, jednoduchý panel nad ním informuje o probíhající operaci a o aktuálním energetickém stavu.



Obr. 5.6 Objekt ST_LEP

5.3.5 Objekt ST_SV (stacionární svařování)

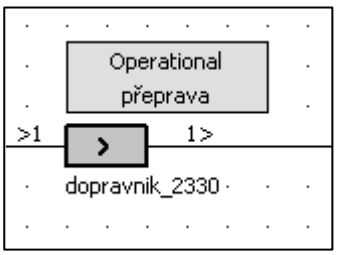
Na lince se, kromě standartního svařování robotem se svařovacími kleštěmi, svařuje i stacionárně. Situace je obdobná jako v případě nanášení lepidla, tedy operaci vykonává robot s griferem, kterým uchopí díl a ten přenese ke stacionární svářečce. Následně dochází k nastavení dílu do požadované polohy a ke spuštění procesu sváření. Po zavaření všech definovaných svářecích bodů robot díl umístí na stůl. Protože je operace velmi podobná stacionárnímu lepení, je modelována stejným způsobem. Pro prvotní rozlišení mají objekty pro stacionární operace šedou barvu, naopak běžné svařování má barvu objektu oranžovou.



Obr. 5.7 Objekt stacionárního svařování

5.3.6 Objekt DOP (dopravník)

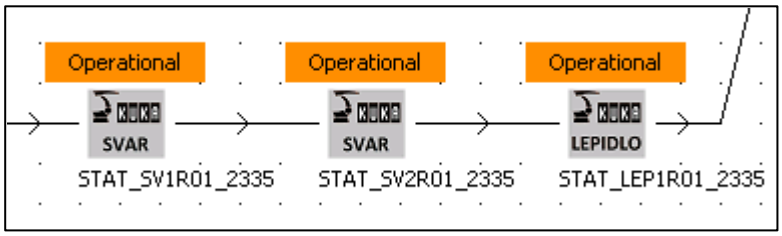
V tomto případě se jedná jen o lehkou úpravu původního objektu a to především ve snaze sjednotit vzhled objektů a jejich informačních panelů. Žádná další přidaná funkce zde není.



Obr. 5.8 Objekt dopravníku

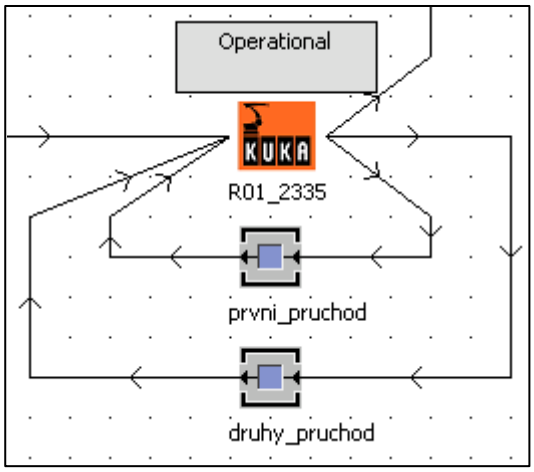
5.3.7 Objekt ROB (robot vykonávající více operací)

V první fázi návrhu linky se počítalo s využitím doteď nedefinovaných vlastních objektů, samozřejmě v kombinaci s objekty ze základní knihovny. Pokud robot na lince vykonává jen jednu operaci, je situace jasná a bezproblémová. Avšak, v rámci zvýšení efektivity a produktivity je zde několik robotů, které vykonávají postupně několik operací, například nejdřív vaří a následně díl přesunou na další stanoviště. Kombinace těchto operací byla nejdříve modelována jako posloupnost již nadefinovaných objektů pro svařování, lepení a přenos dílů. Tento postup se zdál být nejjednodušší a nejpřehlednější. Celá linka byla vyobrazena jako posloupnost operací a na první pohled bylo možno vypozorovat i materiálový tok, tedy odkud kam díl směřuje. Problémem, který se projevil již při modelování objektu stacionárního svařování a lepení, byla možnost zpracovávání více dílů jedním robotem v jednom okamžiku. Díky zvolenému přístupu totiž objekty nepředstavovaly fyzické stroje a zařízení, ale jejich operace. Pokud zařízení vykonávalo jen jednu operaci, bylo vše v pořádku. Nicméně pokud robot vykonával operací více, které navíc nemusely následovat bezprostředně po sobě, docházelo k tomu, že v modelu jeden robot pracoval na více místech a v jeden okamžik vykonával více operací s několika díly. Aby se tomuto zabránilo, bylo potřeba zablokovat vstup do objektu první operace robota, dokud robot vykonává operaci jinou nebo jinou operaci vzhledem k materiálovému toku bude vykonávat ještě předtím, než přijme nový díl ke zpracování. Za pomoci metod, které byly volány právě při vstupu do první operace a výstupu z poslední operace robota, byla vytvořena logika, která výskytu tohoto problému zabraňovala. Tímto způsobem bylo docíleno zprovoznění virtuálního modelu celé linky, který věrně kopíroval výrobní proces.



Obr. 5.9 Modelování za pomoci postupných operací jednoho robota

Důvod k nutnosti vytvoření dalšího objektu *ROB* a změně celkové struktury linky byl odhalen až při analýze vytvořeného modelu. Pokud jsme totiž zkoumali statistiky jednotlivých objektů, dostávali jsme informace o konkrétních operacích, ne však o konkrétních robotech (u robotů vykonávajících více operací). Statistiky operací v tomto případě neměly žádnou vypovídající hodnotu a byly nepoužitelné. Žádnou funkcí sloučení statistik více objektů do jednoho Plant Simulation nedisponuje, musela by být doprogramována ručně, což by bylo značně složité. Navíc, kromě statistik se zde dále objevil problém se sloučením energetických stavů operací a poruch robotů. Proto bylo nakonec přikročeno k vytvoření zcela nového objektu ROB pro simulaci několika nezávislých operací.



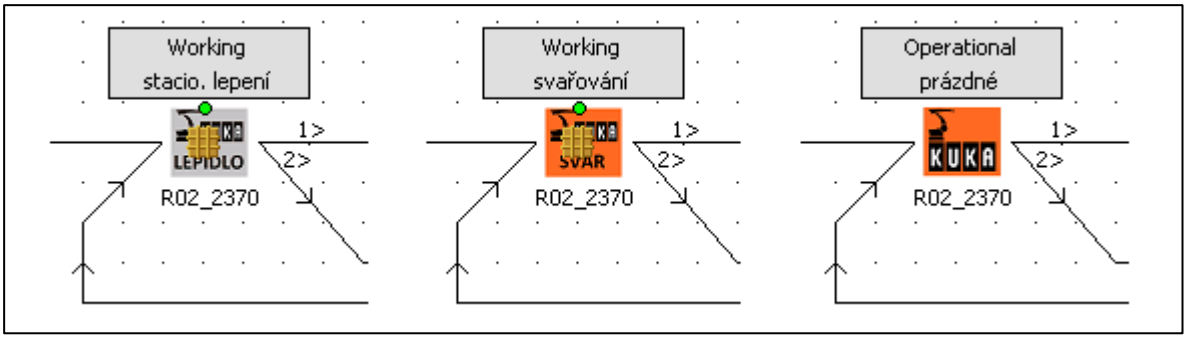
Obr. 5.10 Modelování fyzickými objekty

Objekt v tomto případě již představuje konkrétního robota. Do tohoto objektu je pak vedeno několik „zpětnovazebních" smyček, díky kterým projde materiál prvkem několikrát. I zde je nutno vytvořit určitou logiku, která s objektem spolupracuje. Konkrétně jde o tři metody. Například pro robota *R01_2310* jde o metody *Vs_2310*, *Vys_2310* a *Odblok_2310*, na kterých bude popsán jejich účel.

Metoda *Vs_2310* je volána při vstupu materiálu do objektu. V prvé řadě zablokuje výstup z předchozího objektu, aby robot nemohl přijímat další materiál. Dále stanovuje, která operace bude probíhat. Operace, které robot může vykonávat, jsou nadefinovány přímo v této metodě. Definice zahrnuje nastavení procesního času, mění se zároveň i ikona objektu a stav v připojeném panelu nad objektem, aby bylo vidět, ke které operaci zrovna

dochází. To, která operace bude zvolena, se rozhoduje na základě toho, od kterého objektu aktuální MU přichází (základní atribut MU - PreviousLocation).

Metoda *Vys_2310* je volána vždy po výstupu materiálu z objektu, kromě nastavení defaultní ikony a stavu objektu rozlišuje, zda na daném MU již proběhla poslední operace tohoto robota. Pokud ano, volá se metoda *Odblok_2310*, která odblokuje dočasně zablokovaný výstup objektu před robotem, ten tak může přijmout další díl. Metoda *Odblok* je v některých případech volána až objektem následujícím po robotovi, nicméně výsledek je totožný.



Obr. 5.11 Stejný objekt ROB vykonávající různé operace

Kromě metod je za výstupní strategii objektu *ROB* vybrána cyklická s blokováním. To zajistí, aby každý MU prošel objektem tolikrát, kolik operací na něm má daný robot vykonat, teprve potom bude pokračovat linkou dále.

Vytvoření prvku *ROB* již umožňuje bez dalšího zásahu rovnou sledovat statistiky robota, kterého má objekt představovat. Stejně tak energetické stavy a poruchy definujeme jednoduše jen v rámci jednoho objektu. Nevýhodou je lehká ztráta přehlednosti o materiálovém toku, kdy nemusí být na první pohled patrné, kam daný MU bude dále směřovat. Nicméně samotná simulace tím zasažena není.

5.4 Vytváření modelu

Vytvořený model v Plant Simulationu se drží stejné koncepce jako model v Process Designeru. Hlavní frame nese označení *Linka* a zobrazuje 3 oddělená pracoviště BMS1,

BMS2 a BMS3 (Plant Simulation neumožňuje, aby název objektu začínal číselnou hodnotou, proto je zde číslo na rozdíl od původního projektu na konci názvu). Každé pracoviště je tvořeno opět svým vlastním framem, ve kterém jsou modely výrobních buněk, v případě BMS2, které má pouze jednu buňku, je zde již model této části linky.

Objekty vytvořených pracovišť jsou pro přehlednost graficky individualizovány. Kromě toho můžeme na hlavním framu vidět *EventController* řídící průběh simulace, zdroje veškerého materiálu (jednotlivých dílů) pro všechny pracoviště, objekty pro vykreslení grafů, nastavení směn a několik metod, které budou postupně popisovány v dalších podkapitolách.

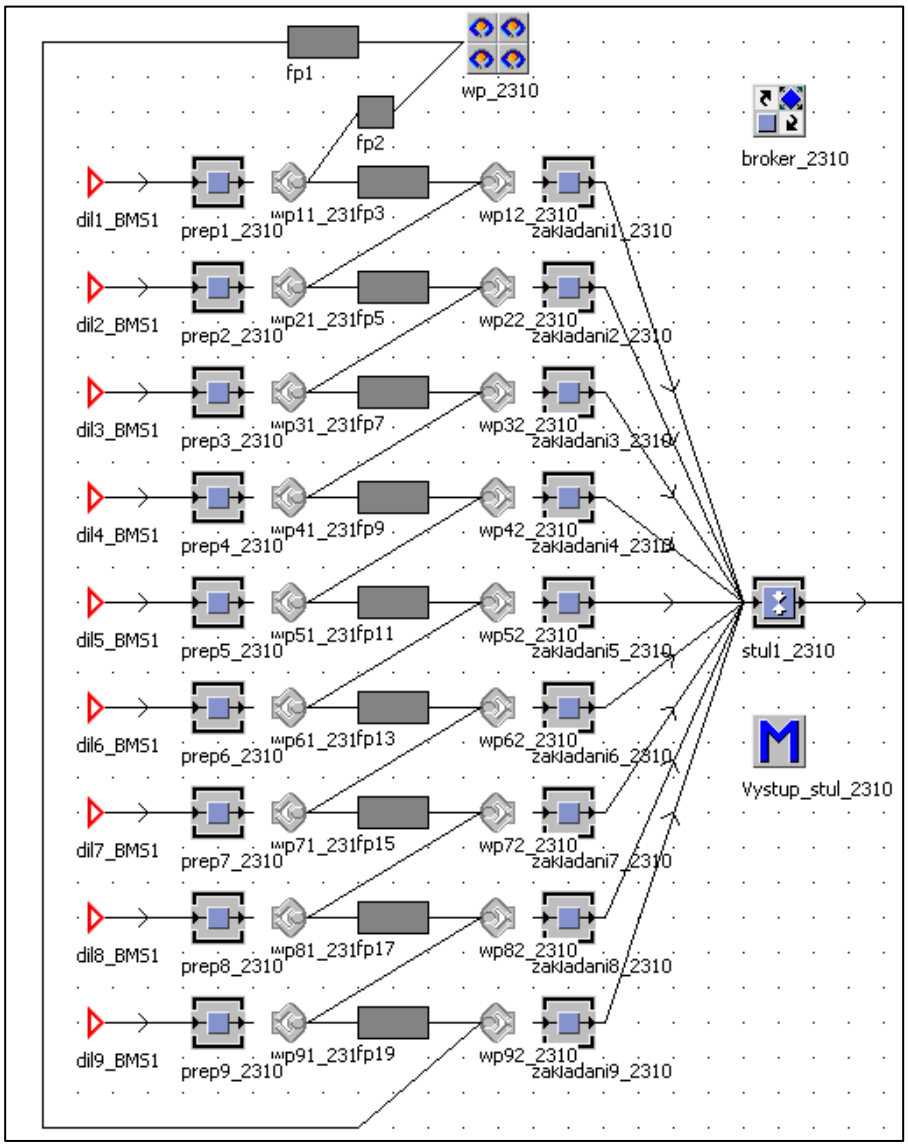
Metoda *Init*, která se nachází i ve framu u každého koncového modelu, slouží k inicializaci několika parametrů, jako odblokování všech zařízení a jejich přepnutí do energetického stavu Off před zahájením samotné simulace. *Init* se spouští automaticky při spuštění simulace, avšak ještě předtím, než se rozběhne čas.

Postup modelování a vytvořená logika je z důvodu úspory textu detailně popsána pouze u SK1 pracoviště BMS1. Ostatní části linky pracují se stejnými objekty, jen v modifikované struktuře, ale chování objektů, metod a materiálového toku je shodné, popisovat je by tak bylo jen opakováním již napsaného. Struktura a návaznost objektů v každé z podčástí vychází ze sekvence probíhajících procesů uvedené v příloze 9.1.

5.4.1 BMS 1 – SK 1

Vstupní bod modelu je tvořen objekty *Interface*, na které jsou napojeny zdroje z framu *Linka*. Následují objekty *prepX_2310* (kde X je číslo v rozsahu 1 až 9), které znázorňují místo pro odběr materiálu obsluhou. Z toho důvodu je ke každému tomuto objektu přiřazen objekt *Workplace* s názvem *wpX1_2310*. V reálném prostředí jsou díly zakládány na jeden stůl, avšak každý díl má na stole své konkrétní místo. Toto místo představují zakládací stanice s názvem *zakladaniX_2310*. Aby sem mohl být díl ručně vložen, je potřeba i zde přidat ke každému objektu *Workplace*, nyní s označením *wpX2_2310*. Pracoviště pro ruční odebrání a založení dílu jsou propojeny za pomoci cest (*FootPath*), které se připojují, tak jako všechny objekty v modelu, standardním objektem *Connector*. Délka cest je stanovena podle dodaných časů zakládání. Pracovník má pro zjednodušení

definovánu rychlost pohybu 1 m/s, cesty mají délku 2 metry. Aby časy přesně odpovídaly, jsou doladěny stanovením doby potřebné k odebrání a založení dílu v objektu *Workplace*. Po založení dílu do zakládací stanice se pracovník přesouvá k další stanici pro odebrání dílu. Po založení posledního dílu se vrací do objektu *wp_2310* typu *WorkerPool*, kde čeká na další příkazy.

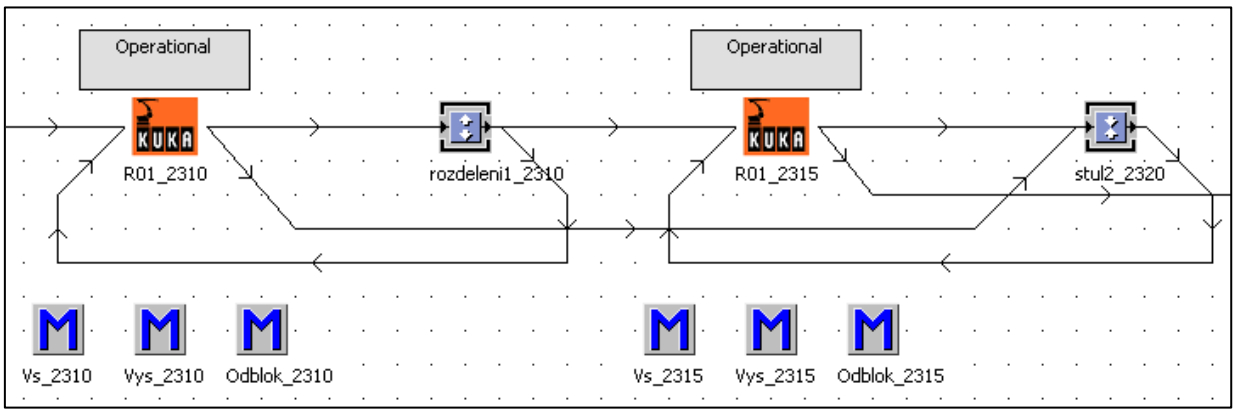


Obr. 5.12 BMS 1 – SK 1: Simulace ručního zakládání dílů

K objektu *wp_2310* je přiřazen *Broker* s názvem *broker_2310*, který zajišťuje obsluhování stanic vysláním pracovníka, který tuto obsluhu vykonává. Každý objekt *prepX_2310* má za výstupní strategii zvolenu Carry Part Away, která říká, že materiál je ze stanice odnášen ručně, nikoli běžným výstupem objektu. Kromě stanovení výstupní strategie musíme

vybrat, kam má být daný díl přesunut, jaký *Broker* zpracovává požadavek na obsluhu stanice a která servisní operace je vyžadována. Pro tento účel byla vytvořena servisní operace *zakladani_2310*. Aby ji mohl vykonávat pracovník vygenerovaný ve *wp_2310*, musíme mu ji v atributech přidat do servisních operací, kterých je schopen.

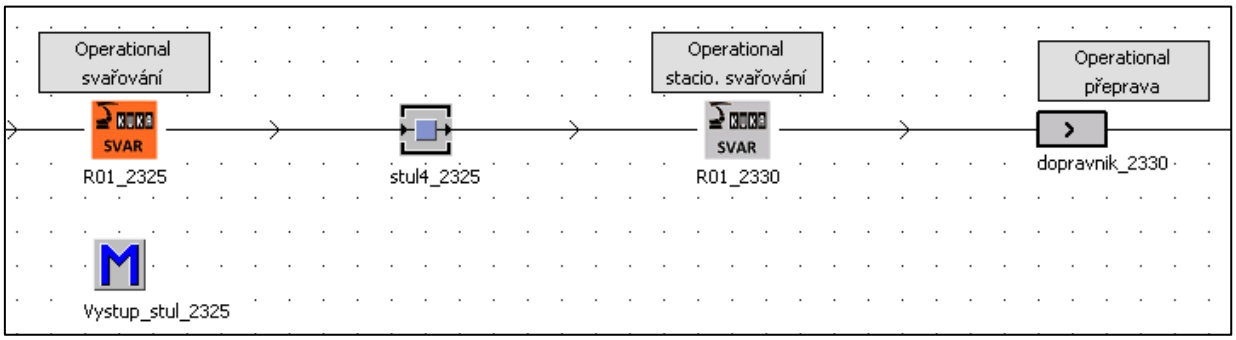
Po ručním založení všech devíti dílů dojde v objektu *stul1_2310* ke spojení v jeden celek. Fyzicky jsou díly samozřejmě stále oddělené, nicméně stůl je již upínkami drží při sobě a následující objekty pro svařování umí pracovat pouze s jedním MU, proto toto sjednocení. *stul1_2310* má nastaven nenulový procesní čas, který symbolizuje nastavení stolu do požadované polohy a sevření upínek.



Obr. 5.13 BMS 1 – SK 1: Robotické operace po založení dílů

Dále následuje objekt *R01_2310* námi definovaného typu *ROB*, což je robot vykonávající nejdříve svařování a následně přenos dílu na další stůl. Díl do objektu vstupuje nejprve větví předchůdce č. 1. Dochází ke svařování, které je indikováno jak na panelu nad objektem, tak i změnou ikony objektu. Po dokončení operace je MU odeslán následníkovi po větvi číslo 1, kde v *rozdeleni1_2310* dochází k rozdělení materiálu na dva. Následně probíhá paralelní operace dvou robotů. Jedna část se vrací do objektu *R01_2310* po větvi předchůdce č. 2 a dochází k přenosu této části na *stul2_2320*. Druhá část vstupuje do objektu *R01_2315* po větvi předchůdce č. 1 a dochází zde k operaci stacionárního nanášení lepidla. Robot *R01_2315* poté umisťuje díl také na *stul2_2320*, tím dojde ke slepení obou částí a tedy i k opětovnému sloučení v jeden MU. Po tomto spojení se díl dostává opět do objektu *R01_2315* (větev předchůdce č. 2), kde nyní robot vykonává operaci přesunu na *stul3_2325*. Na tomto stole se svařuje, což je znázorněno přesunem

MU do *R01_2325*. Po dokončení je materiál posunut na *stul4_2325*. Protože se fyzicky jedná o jeden a ten samý stůl, jsou zde metody, které ho blokují po celou dobu, co je obsazený, jejich přesná funkce je popsána dále. I zde mají oba objekty pro stůl nenulový procesní čas, který symbolizuje čas nutný pro otevření a uzavření upínek.



Obr. 5.14 BMS 1 – SK 1: Další robotické operace a přenos na dopravník

Posledními objekty buňky SK1 jsou *R01_2330*, což je robot, který svařuje na stacionární svářečce, a *dopravnik_2330*, na který se umístí zhotovený polotovar. *dopravnik_2330* posunuje, za pomoci objektu *vystup_SK1_BMS1* typu *Interface*, MU do SK2 pracoviště BMS1. Dopravník má zároveň i zásobníkový charakter a jeho maximální kapacita je 2 MU.

5.4.2 Metody

Následující metody ošetřují průběh simulace tak, aby probíhala realisticky a věrně kopírovala procesy dodaného modelu linky. Pokud to lze, jsou příslušné metody umístěny vždy v blízkosti objektu, ke kterému se vztahují. Metody jsou popisovány na konkrétních příkladech, nicméně některé se zde vyskytují vícekrát, jen pracují s jiným objektem, funkčnost je ale totožná.

5.4.2.1 Metoda Init

Metoda pro inicializaci simulace, spouští se těsně před spuštěním času simulace. Její funkce spočívá v odblokování všech vstupů a výstupů blokovaných zařízení, pro roboty s více operacemi navíc nastavuje počáteční zobrazení na informačním panelu.

5.4.2.2 Metoda Prubeh

Je volána objektem *t_Prubeh*, který metodu spouští v nekonečné smyčce s intervalem 0,1 vteřiny. Zajišťuje správné hodnoty atributů, které se v čase mění podle stavu simulace. Pro námi popisované pracoviště se konkrétně jedná o zablokování zakládacích stanic za podmínky, že je *stul_1_2310* obsazen.

Dále se zde nastavuje zobrazení informačních panelů pro roboty, které vykonávají pouze jednu operaci. Pokud je objekt robota obsazen, indikuje se probíhající operace, pokud robot nepracuje, zobrazuje se, že je neobsazen.

```
(pred : boolean; po : boolean)
is
do
    if stul1_2310.Full then
        zakladani1_2310.EntranceLocked := true;
        zakladani2_2310.EntranceLocked := true;
        zakladani3_2310.EntranceLocked := true;
        zakladani4_2310.EntranceLocked := true;
        zakladani5_2310.EntranceLocked := true;
        zakladani6_2310.EntranceLocked := true;
        zakladani7_2310.EntranceLocked := true;
        zakladani8_2310.EntranceLocked := true;
        zakladani9_2310.EntranceLocked := true;
    end;


    if not R01_2325.Empty then
        R01_2325.druhOperace := "svařování";
    else
        R01_2325.druhOperace := "prázdné";
    end;

    if not R01_2330.Empty then
        R01_2330.druhOperace := "stacio. svařování";
    else
        R01_2330.druhOperace := "prázdné";
    end;

    if not dopravnik_2330.Empty then
        dopravnik_2330.druhOperace := "přeprava";
    else
        dopravnik_2330.druhOperace := "prázdné";
    end;


end;
```

Obr. 5.15 BMS 1 – SK 1: Zdrojový kód metody Prubeh

5.4.2.3 Metody Vs_2310, Vys_2310 a Odblok_2310

Tyto metody řídí materiálový tok objekty, především robotů, které zpracovávají více operací. Jejich funkce byla popsána již při definici objektu *ROB* v podkapitole 5.3.7. Pro robota *R01_2310* se metoda *Vs_2310* volá při každém vstupu MU a blokuje výstup pro předchozí objekt, tedy pro *stul1_2310*. Po druhém průchodu MU objektem *R01_2310* je volána metoda *Vys_2310*, která výstup pro *stul1_2310* opět odblokuje. To za pomoci volání metody *Odblok_2310*. Kromě toho se tyto metody podílejí i na řízení informačního panelu objektu, konkrétně probíhající operace.

5.4.2.4 Metody Vstup_stroj a Vystup_stul_2325

Stůl je objekt, který se v modelu může objevit vícekrát, i když se fyzicky jedná o jeden a ten samý stůl. Proto jsou zde metody *Vstup_stroj* a *Vystup_stul_X*. Uveďme konkrétní příklad pro stůl 2325. Podobně jako v případě robotů vykonávajících více operací, se při vstupu MU do *stul3_2325* metodou *Vstup_stroj* vstup dalších MU do tohoto objektu zablokuje. Teprve po opuštění stolu, tedy prvku *stul4_2325*, je metodou *Vystup_stul_2325* vstup do *stul3_2325* opět povolen. Podobným způsobem funguje i blokování zakládacího stolu *stul1_2310*, který když je obsazen, jsou blokovány i jednotlivé zakládací stanice *zakladani1_2310* až *zakladani9_2310*, protože jsou fyzicky součástí stolu.

5.4.2.5 Metody Vs_wp_2310 a Vys_wp_2310

Aby v budoucnu bylo možné definovat přestávky a konce směn, je nutné zajistit, aby pracovníci po založení všech devíti dílů odcházeli do *WorkPoolu*, tedy aby procházeli objektem *wp_2310*. Zaškrtnutím políčka Get job order in the pool only můžeme pracovníky donutit se sem vracet, v tomto případě se však vracejí po založení každého jednotlivého dílu, výsledná operace by tak měla odlišný charakter a trvala by neúměrně dlouhou dobu. Pokud políčko nezaškrtneme, pracovník zakládá díly postupně, objeví-li se před založením posledního dílu na některém z předcházejících stanovišť díl nový, člověk může obsloužit stanici bez průchodu *WorkPoolem*, protože příkazy pro obsluhu přijímá vzdáleně.

Proto byly vytvořeny metody, které toto defaultní chování upravují. Po vystoupení člověka z *wp_2310* je volána metoda *vys_wp_2310*, která zablokuje vstup do všech stanic

prep1_2310 až *prep9_2310*. Tím je zajištěno, že se nový materiál nebude znovu nikde objevovat, tím pádem nevznikne nový požadavek na obsluhu a pracovník se po odnesení všech dílu vrátí nejkratší cestou do *WorkPoolu*. Jakmile do něho vstoupí, vstup do stanic pro odebírání materiálu se opět odblokuje, aby mohly být přijaty nové MU. Toto odblokování probíhá postupně s 0,1 vteřinovým zpožděním u každého dalšího stanoviště. Pracovník totiž vyřizuje požadavky podle toho, jak je přijal. Mohlo by se tedy stát, že se díly naskladní v jiném pořadí, než námi určeném, což by opět znamenalo rozhození předdefinované operace a nárůst času potřebného pro kompletní založení všech MU. Postupným odblokováním se tomuto problému vyhneme.

5.5 Analýza a experimenty

Nyní budeme zkoumat vlastnosti namodelované linky, její možnosti výroby a vliv několika faktorů na výslednou produkční schopnost a celkovou energetickou spotřebu. Podrobnější grafy k následujícím kapitolám jsou obsahem přílohy 9.4.

5.5.1 Základní analýza

Připravený model můžeme v první fázi ihned odsimulovat. Tím dostaneme informace o rychlosti výroby, počtu výrobků, spotřebě linky a vytížení zaměstnanců. Protože zatím neuvažujeme žádná omezení ve výrobě, jako jsou poruchy robotů, pauzy pro pracovníky, údržba strojů a podobně, můžeme tato prvotní data považovat za určité maximum, čeho jsme na lince schopni dosáhnout. Neuvažujeme zatím ani přechod zařízení do energeticky úsporného stavu. Kromě ověření správnosti výsledků, zda jsou dosažené hodnoty realistické a tím pádem je i náš model správně realizován, dostáváme srovnávací kritéria pro následující experimenty.

Sledování výroby bude probíhat ve dvou podobách, jedna s časem simulace 24 hodin, druhá s délkou jednoho týdne. Největší pozornost budeme věnovat produkci, tedy počtu vyrobených kusů za den, a spotřebě energie. V tabulce 5.4 jsou shrnuty údaje počáteční analýzy, grafy využití zdrojů, jejich spotřeba podle stavů a vytížení zaměstnanců jsou obsahem přílohy 9.4.1.

Jak je vidět v tabulce, v této fázi, především díky absenci směn, není patrný rozdíl mezi týdenní a denní výrobou.

	1 den	1 týden
Celkový počet vyrobených dílů	1221	8543
Průměrná doba průchodu jednoho dílu [min:sec]	13:03,1	13:03,1
Průměrný interval dokončení dílu [min:sec]	01:10,8	01:10,8
Průměrný počet dílů za hodinu	50,88	50,85
Průměrný počet dílů za den	1221	1220,4
Celková spotřeba [kWh]	875,1	6125,7
Spotřeba ve stavu Operational [kWh]	138,6	969,8
Maximální příkon (špička) [kW]	41,4	41,4

Tab. 5.4 Základní analýza

5.5.2 Směny

Aby simulace popisovala výrobu co nejrealističtěji, bude zapotřebí stanovení pracovních směn. Konkrétní časy jsou uvedeny v tabulce 5.5. Linka se rozjíždí ve 22:00 v neděli, běží celý pracovní týden (kromě pauz) a provoz se ukončuje v sobotu v 6:00 ráno.

	Časový interval	Pauza
Ranní	6:00 - 14:00	10:30 - 11:00
Odpolední	14:00 - 22:00	18:30 - 19:00
Noční	22:00 - 6:00	2:30 - 3:00

Tab. 5.5 Definice směn

Testujeme dvě možné strategie. První (označení Hromadná pauza) je, v případě pauzy nebo víkendu, jednoduché opuštění linky pracovníky s tím, že výroba se nechá ve stavu, ve kterém se při odchodu nacházela. Další možnou strategií (označení Postupná pauza) je postupné dokončení výrobků, které se již na lince nacházejí. Pauzy tedy nebudou vymezeny stejně pro celou linku, pouze se v určitý moment přestane zakládat nový

materiál. Pracovníci i stroje tak přestávají pracovat postupně, podle toho, v jaké části linky se nacházejí. Půlhodinová pauza by u všech zaměstnanců měla být i přesto dodržena. Výsledky obou strategií najdeme v tabulce 5.6. Můžeme si všimnout, že nyní jsou mezi denní (pracovní den) a týdenní výrobou již rozeznatelné rozdíly. Obě strategie je nutno brát s určitou rezervou, ani jedna nebude přesně kopírovat chování na reálné lince, nicméně pro námi zkoumané experimenty jsou dostačující.

	Hromadná pauza		**Postupná pauza**	
	1 den	**1 týden**	**1 den**	**1 týden**
Celkový počet vyrobených dílů	1128	6095	1146	6198
Průměrná doba průchodu jednoho dílu [min:sec]	13:59,8	18:38,1	13:14,4	01:37,4
Průměrný interval dokončení dílu [min:sec]	01:15,3	01:39,0	01:14,2	01:37,6
Průměrný počet dílů za hodinu	47	36,28	47,8	36,9
Průměrný počet dílů za den	1128	870,7	1146	885,4
Celková spotřeba [kWh]	838,4	5155,2	845,6	5197,5
Spotřeba ve stavu Operational [kWh]	153,6	1475,1	149,9	1452,9
Maximální příkon (špička) [kW]	41,4	41,4	42,7	42,7

Tab. 5.6 Analýza při směnném provozu

5.5.3 Přechod do energeticky úsporného stavu

Můžeme navrhnout první úsporné opatření, a to uspávat roboty v době, kdy se nachází ve stavu Unplanned. Pro to je nutno vysvětlit, jakým způsobem se do tohoto režimu přepíná.

5.5.3.1 Metody pro přechod do režimu Unplanned

Každý objekt v Plant Simulationu můžeme přepínat mezi 3 stavy, jedná se o Planned, Unplanned a Paused. Pokud je stroj mimo svůj vymezený pracovní čas, během pauzy nebo o víkendu, nachází se ve stavu Unplanned. Paused je v našem konkrétním případě označení pro pozastavení práce z důvodu poruchy jiného zařízení, jak bude dále detailněji popsáno.

Přepnutí do Unplanned podle první strategie (celá linka najednou) zajišťuje metoda *Rizeni*, která je volána každou 0,1 vteřiny. Kontroluje stav objektu *smeny*, který na základě aktuálního času rozlišuje mezi běžným provozem, víkendem nebo pauzou. Pokud má dojít k přepnutí do Unplanned, nejprve se zajistí, aby každý zaměstnanec dodělal svoji práci a vrátil se do svého *Workpoolu*. To se provede zablokováním stanic pro odebírání, kde se tím pádem neobjeví nový díl, pracovník je bez práce, zůstane na místě a může být přepnut do Unplanned. Jakmile jsou všichni pracovníci v tomto stavu, je možné přepnout i všechny zbylé objekty. Po ukončení pauzy opět nastavujeme Planned a odblokujeme odebírací stanice, linka se tak znovu rozjede.

Chceme-li se řídit druhou strategií, je situace o něco komplikovanější. Aby bylo možno dovyrobit veškeré rozdělané díly, je nutno zajistit, aby do všech buněk linky založili pracovníci stejný počet dílů. To má na starosti metoda *Usinani* v hlavním framu *Linka*. Kontroluje stav objektu *smeny*, pokud má dojít k pauze, zjistí maximální počet založených dílů napříč všemi stanicemi, a postupně blokuje zdroje dílů tak, aby se počty v ostatních stanicích vyrovnaly. Jakmile se toto stane, pracovníci nemají co zakládat a zůstávají ve *Workpoolu* - mají pauzu. Materiál postupuje dále linkou a metoda *Usinani*, která je v každém framu jednotlivé výrobní buňky, zajistí přechod do Unplanned u každého zařízení, které již zpracovalo poslední vložený díl před pauzou. To se děje kontrolou zařízení předchozích, jestli se zde nenachází ještě nezpracované díly. Postupně tak dojde k uspání celé linky.

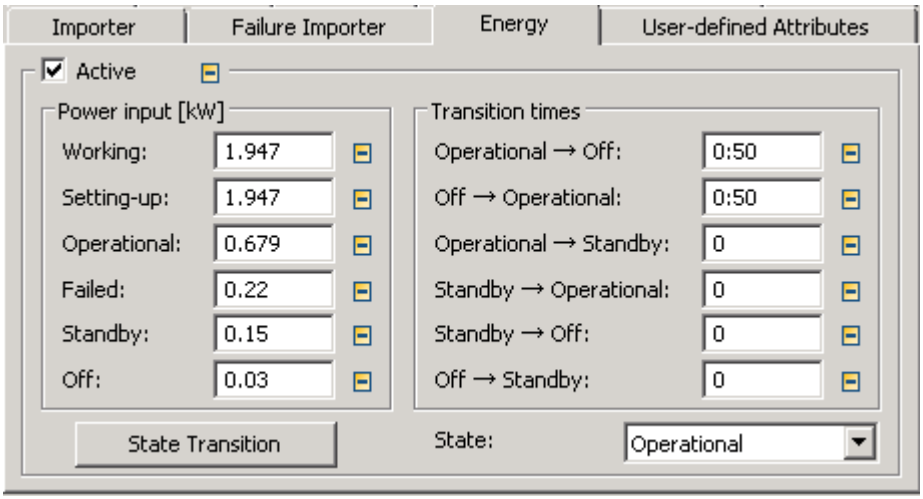
Pro opětovné přepnutí do režimu Planned odblokujeme zdroje materiálu, zbytek má už na starosti metoda *Probouzeni*. Ta funguje jednoduše. Kontroluje objekty, zdali zpracovávají díl, pokud ano, přepne následující objekt po směru materiálového toku do provozního režimu. Takto se ještě chvíli před příchodem dílu uvede zařízení do provozu.

5.5.3.2 Energetické hodnoty spotřeby

Stav Unplanned může souviset i se změnou energetického stavu zařízení. V předešlé simulaci se zavedením směn se energetický režim nechával beze změny, zařízení sice nepracovalo a bylo v Unplanned, ale zároveň energeticky v Operational s k tomu adekvátní spotřebou energie. Plant Simulation nabízí možnost volby, do jakého energetického módu se má objekt přepnout, pokud je v Unplanned nebo Paused. V metodě *Init* se zpočátku simulace definuje, zda má docházet k uspávání robotů, tedy zda

mají v případě stavu Unplanned přejít do energetického režimu Off. V praxi to znamená, že jakmile dojde k pauze během pracovní doby, nebo je právě víkend, zařízení budou v úsporném módu. Při přechodu do Off i zpět je respektován i čas přechodu do tohoto módu, podle tabulky 5.1 v kapitole 5.3.1. Energetická úspora tohoto řešení je z výsledků patrná.

Již v předešlých simulacích pracujeme s hodnotami celkové spotřeby linky, spotřeby ve stavu Operational a maximální špičky příkonu. Každý objekt představující určité zařízení, má v Plant Simulationu svůj jakýsi energetický štítek. Tam nastavujeme spotřeby ve všech dosažitelných stavech. Protože nemáme k dispozici konkrétní hodnoty spotřeb pro jednotlivé operace naší výroby, stanovujeme hodnoty příkonu globálně pro všechna zařízení stejného typu. Předpokládáme, že se na spotřebě modelované linky podílí jen robotické manipulátory a dopravníky.



Obr. 5.16 Energetická spotřeba robota

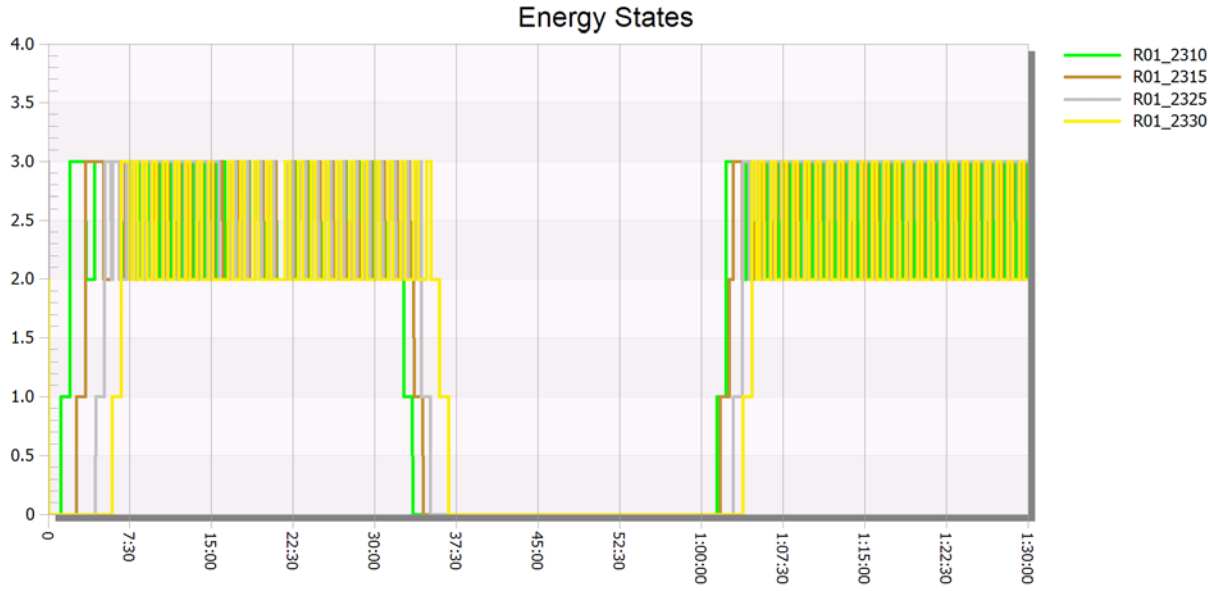
Obrázek 5.16 ukazuje nastavení hodnot pro robota. Pro stanovení příkonu ve stavech Working, Setting-Up a Operational vycházíme z několika měření, která máme k dispozici. Ta sice pochází z jiné linky, kde probíhaly jiné operace, ale vypočtená průměrná hodnota je právě výsledných 1947 W příkonu. Mód Setting-Up symbolizuje energetickou náročnost „nastavování“ robota, u svařování jde typicky o broušení čepiček svařovacích kleští. Protože jde o operaci vykonávající obecný pohyb, příkon je shodný s Working. Zbytek údajů je podle tabulkových hodnot z tab. 5.1. Rozdíl je jen ve stavu Operational,

kde tabulkové hodnoty předpokládají zapnuté brzdy a vypnuté motory. Z měření jsme však dostali příkon roven 679 W, nejprve tedy počítáme s ním.

Rozšířením předešlého experimentu o přepínání do energeticky úsporných stavů dostáváme výsledky podle tab. 5.7.

	Hromadná pauza		Postupná pauza	
	1 den	**1 týden**	**1 den**	**1 týden**
Celkový počet vyrobených dílů	1126	6085	1139	6162
Průměrná doba průchodu jednoho dílu [min:sec]	14:03,1	18:18,3	13:17,3	01:37,9
Průměrný interval dokončení dílu [min:sec]	01:15,5	01:39,2	01:14,6	01:37,9
Průměrný počet dílů za hodinu	46,9	36,2	47,5	36,7
Průměrný počet dílů za den	1126	869,3	1139	880,3
Celková spotřeba [kWh]	815,9	4412,1	826	4470,5
Spotřeba ve stavu Operational [kWh]	130,9	700,3	133,2	713,6
Maximální příkon (špička) [kW]	41,4	41,4	43	43

Tab. 5.7 Změna spotřeby a produkce při uspávání zařízení



Obr. 5.17 Graf postupného uspávání a probouzení robotů pro stanici BMS 1 – SK 1

(0 – Off, 1 – PoweringUp/Down, 2 – Operational, 3 – Working)

Vzhledem k vyšší produktivitě budeme v dalších experimentech pracovat jen s variantou postupného uspávání. Energetická spotřeba je sice mírně vyšší, ale v tuto chvíli je pro nás důležitější počet vyrobených dílů. V reálném provozu mohou být požadavky samozřejmě jiné. Jelikož je tato výrobní linka malou součástí celé svařovny, bude nejdůležitějším faktorem plnění denního plánu vyrobených dílů, aby nedocházelo k přílišnému hromadění zásob. My však v rámci této práce bereme linku jako samostatnou a nezávislou výrobní buňku.

5.5.4 Přechod na brzdy robota

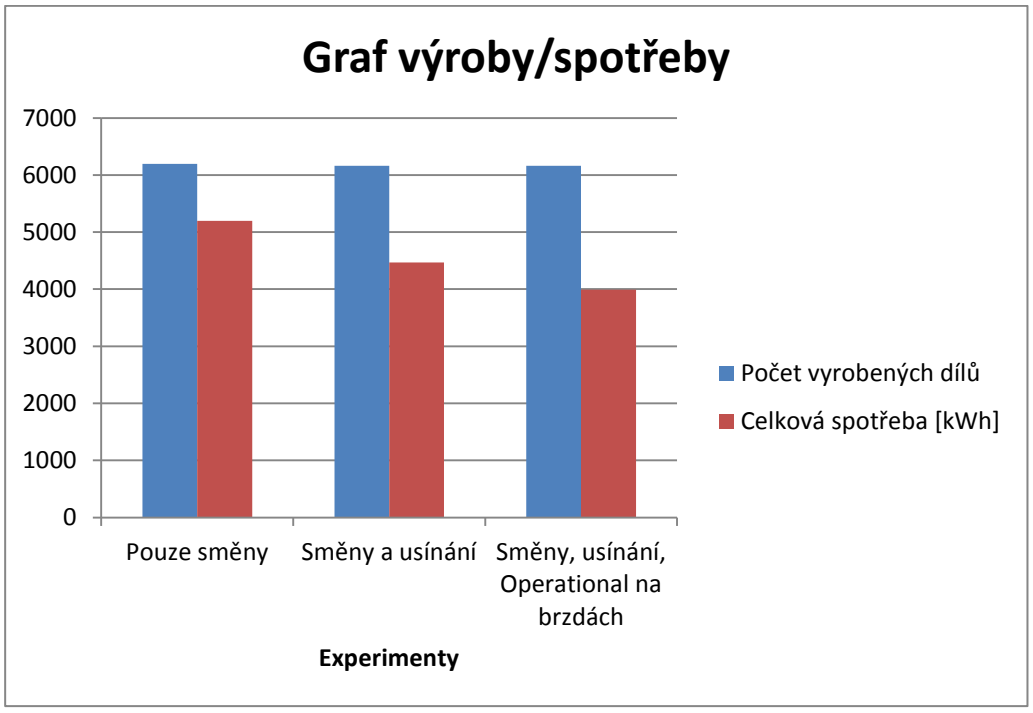
Další, a na realizaci, alespoň teoreticky, poměrně jednoduché řešení pro úspornější provoz je co nejrychlejší přepnutí robotů na brzdy. Jak již bylo uvedeno, v minulém experimentu jsme počítali s reálně naměřenou hodnotou. Nyní budeme uvažovat spotřebu v režimu Operational podle tabulky 5.1, konkrétně 220 W namísto původních 679 W. Tímto opatřením ušetříme energii v momentě, kdy robot během provozu čeká na další díl, ale zároveň zde není dostatečný prostor pro jeho uspání. Tato úspora je pouze orientační, v reálných aplikacích se samozřejmě brzd u robotů využívá. Přímo v kontroléru robota lze nastavit čas, po kterém se robot na brzdy po dokončení operace přepne. My v tomto bodě předpokládáme ideální případ, tedy spuštění brzd ihned po skončení poslední operace. Úspora ve stavu Operational je v tabulce 5.8 patrná na první pohled.

	1 den	**1 týden**
Celkový počet vyrobených dílů	1139	6162
Průměrná doba průchodu jednoho dílu [min:sec]	13:17,3	01:37,9
Průměrný interval dokončení dílu [min:sec]	01:14,6	01:37,9
Průměrný počet dílů za hodinu	47,5	36,7
Průměrný počet dílů za den	1139	880,3
Celková spotřeba [kWh]	736,5	3991,1
Spotřeba ve stavu Operational [kWh]	43,7	234,2
Maximální příkon (špička) [kW]	41,6	31,6

Tab. 5.8 Redukce spotřeby při přepnutí robota na brzdy

5.5.5 Dosažené úspory

Výslednou spotřebu při zavedení všech úsporných opatření, která byla popsána výše, srovnáme s původním stavem, kdy jsme linku simulovali jen se směnným provozem. Dostaneme tak možnou úsporu, které jsme na lince schopni dosáhnout. V úvahu bereme opět pouze variantu s postupným usínáním při pauzách za celý týden. Výsledek je třeba interpretovat s jistou rezervou, původní stav například nepředpokládá ani vypnutí robotů během víkendové pauzy, ke kterému v reálu takřka určitě dochází. Dosažená úspora je 23,2 % při pouze 0,6 % úbytku vyrobených dílů. Z původního poměru 1,19 dílu/kWh se dostáváme na 1,54 dílu/kWh, z tohoto pohledu jde dokonce o zlepšení o 29,5 %.



Obr. 5.18 Graf úspory energie vzhledem k výrobě

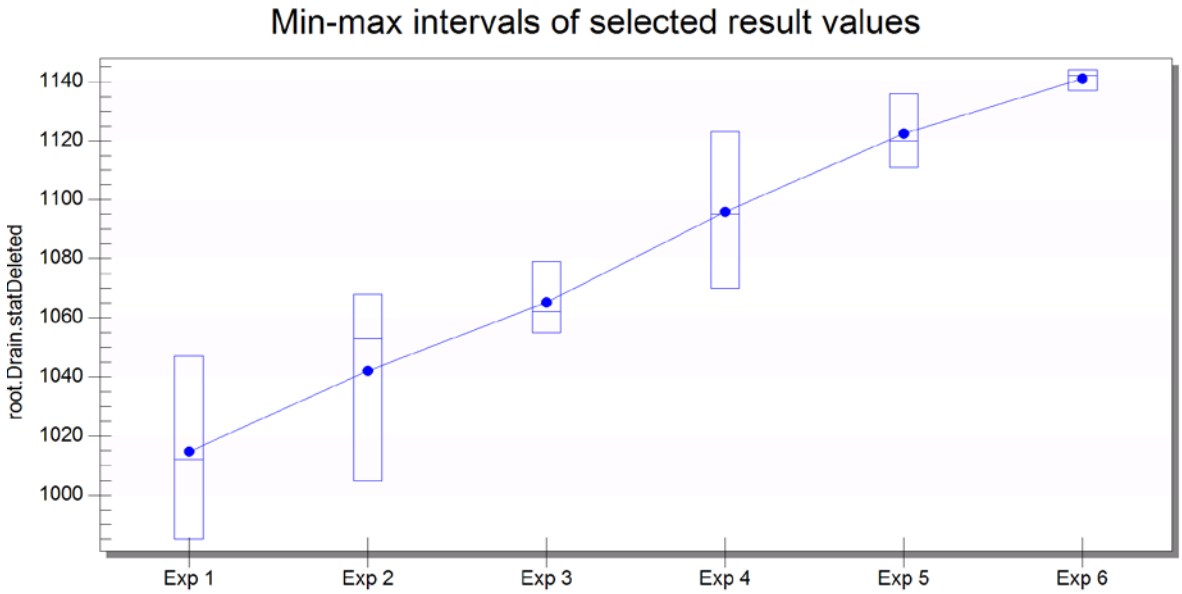
5.5.6 Poruchy

V rámci simulace můžeme jednoduše otestovat chování v případě poruchy zařízení. Stejně jako spotřeby je definujeme v metodě *Init*, která je přednastaví před samotným začátkem simulace. Poruchy můžeme zadat pro každý objekt individuálně, nejjednodušeji přes dostupnost a MTTR. Protože konkrétní hodnoty poruchovosti zařízení nemáme

k dispozici, podíváme se na to, jaký vliv má poruchovost na produkci. Vytvoříme sérii experimentů, kde budeme postupně měnit dostupnost robotů. MTTR necháme na konstantní hodnotě 5 minut. Pro vytvoření experimentů využijeme *ExperimentManager*. Vzhledem k časové náročnosti simulace budeme sledovat vývoj vždy pouze v rámci jednoho dne. Naším cílem bude denní produkce minimálně 1000 výrobků s určitou rezervou. Experiment spustíme pro každý parametr několikrát, protože se poruchy projevují náhodně, více průběhů nám zajistí vyšší statistickou významnost výsledku. Výsledky jsou shrnuty v tabulce 5.9 a vyobrazeny na grafu na obrázku 5.19.

Experiment	Dostupnost	Vyrobeno	Standardní odchylka	min	max
Exp 1	99,0	1014,7	31,1	985,0	1047,0
Exp 2	99,2	1042,0	32,9	1005,0	1068,0
Exp 3	99,4	1065,3	12,3	1055,0	1079,0
Exp 4	99,6	1096,0	26,5	1070,0	1123,0
Exp 5	99,8	1122,3	12,7	1111,0	1136,0
Exp 6	99,9	1141,0	3,6	1137,0	1144,0

Tab. 5.9 Vliv poruch na počet vyrobených dílů



Obr. 5.19 Vliv poruch na počet vyrobených dílů

I při porouchání zařízení můžeme šetřit energii. Metoda *Rizeni*, mimo jiné, zajišťuje, že v případě poruchy minimálně jednoho objektu přejdou ostatní zařízení do stavu Paused. Mód Paused můžeme následně spárovat s akcí na přechod do energetického režimu Off, podobně jako v případě Unplanned.

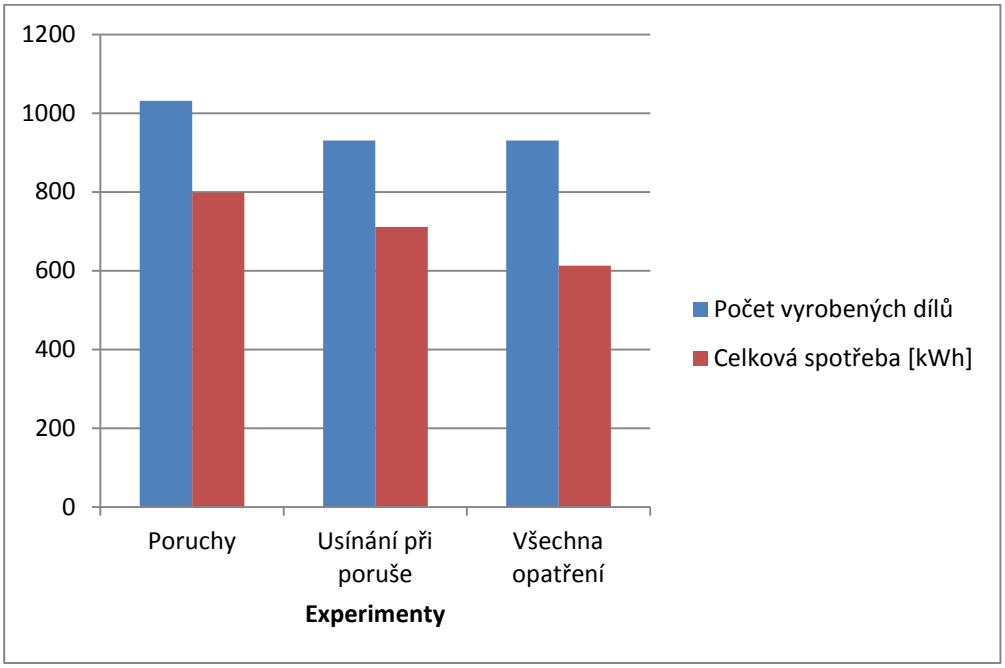
Pro tento experiment stanovíme konstantní poruchovost, a to na základě předchozího experimentu. Tam bylo podmínkou dosažení produkce alespoň 1000 ks/den. Aby zde byla určitá rezerva, volíme dostupnost zařízení rovnu 99,2%. MTTR zůstává rovno 5 minutám. Test provedeme ve třech fázích. První bude počítat pouze se směnným provozem a s nastavenými poruchami zařízení. V druhé fázi přidáme funkci usínání ostatních zařízení během poruchy jednoho ze strojů na lince. Poslední test (všechna opatření) bude kombinovat usínání při poruše včetně všech úsporných opatření, která byla popsána v kapitolách 5.5.3 a 5.5.4. Údaje ze simulací jsou setříděny v tabulce 5.10.

	Poruchy	**Usínání při poruše**	**Všechna opatření**
Celkový počet vyrobených dílů	1031	931	931
Průměrná doba průchodu jednoho dílu [min:sec]	14:33,1	16:18,3	16:18,3
Průměrný interval dokončení dílu [min:sec]	01:22,5	01:30,9	01:30,9
Průměrný počet dílů za hodinu	42,96	38,79	38,79
Průměrný počet dílů za den	1031	931	931
Celková spotřeba [kWh]	798,3	711,1	612,8
Spotřeba ve stavu Operational [kWh]	171,1	141,9	43
Maximální příkon (špička) [kW]	44	42,7	40,9

Tab. 5.10 Úspora během poruch, jednodenní simulace

Z výsledků je patrné, že k úspoře během poruch skutečně dochází, i když za cenu snížení produkce. Srovnáme-li poměr vyrobených dílů na jednu kilowatthodinu, pro samotné poruchy dostáváme 1,29 a pro poruchy s usínáním 1,31. Jedná se o minimální zlepšení, jehož aplikace by se díky menšímu počtu vyrobených dílů v praxi nevyplatila. Přidáme-li úsporná opatření z předešlých částí, dostáváme se na poměr 1,52 dílu/kWh. Logicky, uspávání robotů a ostatních zařízení během doby, kdy nepracují, nemá na produkci samotnou žádný vliv. Je však třeba si uvědomit, že měření probíhalo za určité striktně dané poruchovosti a po poměrně krátký čas. Metoda úspory při poruchách by zcela jistě

byla výhodnější pro poruchy s vyšším MTTR. Další možností je měnit čas, který musí uplynout od poruchy po rozhodnutí o uspání. Krátkodobé poruchy by nezpůsobily přechod do úsporného režimu a výroba by se tím nezdržovala, samozřejmě opět za cenu o něco vyšší spotřeby. Tyto experimenty by nebylo těžké odsimulovat, ale modelování poruch způsobovalo zablokování modelu linky, výsledky proto nebyly použitelné. Pravděpodobná příčina tohoto problému souvisí s doprogramovanou logikou linky, která počítala s určitým materiálovým tokem. Vlivem poruch a kombinací přechodů do různých stavů byl však tento tok narušen a nastala určitá souhra faktorů, která způsobila uváznutí dílu v některém z objektu. Samotná příčina nebyla objevena, i tak lze odhadovat, že její odstranění by vyžadovalo kompletní modifikaci struktury linky, včetně metod.



Obr. 5.20 Graf úspory energie během poruch

6 PROCESS SIMULATE A PLANT SIMULATION

Přestože jsou oba uvedené programy součástí jednoho softwarového balíku, jejich vzájemné možné propojení je minimální. Vývoj každého z nástrojů probíhal odděleně, což je patrné již z grafického uživatelského rozhraní a ovládání. Díky tomu nebyla implementována žádná funkce, která by alespoň do budoucna umožňovala sdílení a importování projektů mezi sebou. Problémem je také zcela odlišná architektura, na které jsou programy postaveny. Výrobce sám tvrdí, že určitá možnost importu projektu z Process Simulate do Plant Simulation zde je, ale jeho možnosti jsou tak omezené, že se vůbec nepoužívá, není proto ani uváděn v žádných materiálech. Jediným řešením je tak vytvoření celého modelu v Plant Simulationu od základů, tedy stejný přístup, jaký byl zvolen i v této práci.

Process Simulate přistupuje k modelování jiným způsobem. Dá se říci, že se jedná o grafický nástroj rozšířený o množství funkcí, které dokážou objekty rozpohybovat a vytvářet mezi nimi závislosti. Simulovanou operaci pak vidíme tak, jak bude probíhat v reálném prostředí, k jakým konkrétním pohybům zde bude docházet a co všechno se s objekty stane. Plant Simulation celou, mnohdy složitou a na modelování náročnou, studii této jedné operace zjednoduší na pár atributů, jako je například čas procesu a jeho energetická náročnost. Dochází tu k ohromné ztrátě informací, ale to je dáno odlišným zaměřením programů. Process Simulate má testovat proveditelnost konkrétních operací, kdežto Plant Simulation zkoumá návaznost již otestovaných a fungujících operací. Dokáže se tak v rámci jednoho modelu soustředit na celou továrnu, a to Process Simulate nedovede, ten ověřuje jednotlivá pracoviště zvlášť. Z tohoto stručného popisu vyplývá, že importování projektu z jednoho programu do druhého je zbytečné, oba se sice zaměřují na simulace, ale každý ji chápe v jiném smyslu. Výsledkem importu rozsáhlé studie z Process Simulate do Plant Simulation může být 10 objektů typu SingleProc, stejně tak jako jeden jediný. Záleží na aplikaci a na tom, co zrovna chceme simulovat.

7 ZÁVĚR

Cílem této práce bylo zasvětit čtenáře do simulování výrobních procesů v rámci jejich virtuálních modelů. Za tímto účelem byla stručně nastíněna práce s programem Plant Simulation, popsány jeho klíčové vlastnosti a možnosti vytvářených modelů. Nabyté znalosti z této teoretické části byly posléze využity při vytváření vlastní simulační studie, na které se zároveň povedlo otestovat sérii experimentů. Přestože vzorem modelu byla reálná linka, je třeba brát výsledky simulace s rezervou. To je dáno jak nedostatkem informací o samotném výrobním zařízení, tak i několika zjednodušením a předpokladům, kterých jsme se však, díky zaměření této práce, mohli dopustit. Nicméně na reálné lince by zde uvedených úspor bylo dosaženo jen stěží, už jen kvůli tomu, že některé úspory jsou již nyní pravděpodobně aplikovány. I přes to má prováděná simulace nepochybně své přínosy, předně díky zvoleným a testovaným postupům, které k úsporám energie určitě vedou.

Co se týče samotného programu Plant Simulation, jeho základní funkčnost je na první pohled poměrně jednoduchá, ale díky jazyku SimTalk jsou jeho možnosti téměř neomezené. S tím souvisí i oblast, pro kterou je tento software určen. Zadanou výrobní linku bylo možno programem namodelovat a vzápětí odsimulovat, nicméně doladění správného chování, kterého bylo docíleno právě doprogramováním logiky, bylo poměrně složité a časově náročné. Simulovat tímto způsobem celou továrnu by bylo nereálné. Z toho vyplývá, že program je zaměřen spíše na celou výrobu nebo na její větší část, než na konkrétní jedno pracoviště. V rámci továrny je pak toto pracoviště modelováno jako pouze jeden objekt a Plant Simulation plánuje a optimalizuje celkovou výrobu až po konečný produkt, a to především z logistického hlediska.

8 ZDROJE

[1] Bangsow, Steffen. Manufacturing Simulation with Plant Simulation and SimTalk. Springer, 2010

[2] Tecnomatix Plant Simulation 10 Step-by-Step Help. Siemens PLM, 2010

[3] Tecnomatix Plant Simulation help

[4] Tecnomatix Plant Simulation Basics, Methods and Strategies Student Guide. Siemens PLM, 2012

[5] Baumruk, Martin. Process Simulate Basic, Modeling&Kinematic. Siemens, 2014

[6] Baumruk, Martin. Process Simulate Advanced, CEE. Siemens, 2013

[7] Baumruk, Martin. Process Simulate Robotic Spot. Siemens, 2014

[8] Baumruk, Martin. Process Simulate Robotic KUKA (VKRC), OLP Basic. Siemens, 2014

[9] Pavlík, Vojtěch. PROFIenergy. ČVUT, Fakulta elektrotechnická, Katedra řídicí techniky, 2014

[10] Urban, Tomáš. Postup pro naimportování dodaných dat do Process Simulate. ČVUT, Fakulta elektrotechnická, Katedra řídicí techniky, 2014

[11] Internetové stránky a firemní materiály společnosti Siemens PLM http://www.plm.automation.siemens.com/cz_cz/

[12] Prezentace, Tecnomatix Process Designer. Technická univerzita v Liberci, Fakulta strojní, 2012

[13] Prezentace, Optimalized Energy Efficiency with Tecnomatix. Matthias Heinicke, Siemens PLM, 2013

[14] Manuál, KUKA System Software 8.2 – Návod k obsluze a programování pro konečné uživatele, KUKA Roboter GmbH, 2012

[15] Siemens PLM Academic Learning Advantage

https://training.plm.automation.siemens.com/mytraining/login.cfm?

9 PŘÍLOHY

9.1 Sekvence operací linky podle stanovišť

1 BMS – 1 SK

Stanice 2310 - ruční zakládání dílu + robotická manipulace

- ruční zakládání 9 dílů přímo na stůl
- odchod obsluhy
- svařování robotem R01_2310
- otočení nástroje o 180 stupňů
- přenos dílu robotem R01_2310 na další stůl
- uchopení 2 částí robotem R01_2315 a přenos k lepící pistoli

Stanice 2315 - robotické nanášení lepidla, manipulace se sestavou

- nanášení lepidla na dvou pistolích robotem R01_2315
- přenos dílu k dalšímu stolu robotem R01_2315

Stanice 2320 - sestavovací stanice

- založení dílu robotem R01_2315
- otočení nástroje o 180 stupňů
- přenos dílu robotem R01_2315 na další stůl

Stanice 2325 - robotické zakládání a svařování

- svařování robotem R01_2325

Stanice 2330 - robotické přivařování matic

- uchopení dílu robotem R01_2330 a přenos ke svařovací stanici
- přivařování matic na stacionární stanici robotem R01_2330
- přenos dílu na dopravník robotem R01_2330

1 BMS – 2 SK

Stanice 2335 - ruční zakládání + robotická manipulace, navaření čepů

- ruční zakládání 2 dílů na dopravníkový přípravek
- uchopení obou dílů robotem R01_2335 a přenos ke svařovací stanici
- přivařování čepů na dvou stanicích robotem R01_2335
- přenos dílu klepící pistoli robotem R01_2335
- nanášení lepidla robotem R01_2335
- přenos dílu robotem R01_2335 na další stůl

Stanice 2340 - robotické svařování

- založení dílu robotem R01_2335
- uchopení a přenos dílu z předchozího pracoviště robotem R01_2340
- založení dílu robotem R01_2340
- otočení nástroje o 180 stupňů
- současné svařování robotem R01_2340 a R02_2340
- uchopení dílu robotem R01_2345 a přenos ke svařovací stanici

Stanice 2345 - robotické svařování

- svařování na stacionární stanici robotem R01_2345
- frézování čepiček na svařovací stanici
- přenos dílu robotem R01_2345 na další stůl

Stanice 2350 - robotické přivařování čepů

- založení dílu robotem R01_2345
- svařování robotem R01_2350
- uchopení dílu robotem R01_2355

Stanice 2355 - robotické svařování

- přenos dílu ke svařovací stanici robotem R01_2355
- svařování na stacionární stanici robotem R01_2355
- frézování čepiček na svařovací stanici
- přenos dílu robotem R01_2355 na další stůl

2 BMS – 1 SK

Stanice 2360 - ruční zakládání dílu + robotické svařování

- ruční zakládání 5 dílů na dopravníkový přípravek
- natočení osy, uchopení 3 dílů robotem R01_2360
- natočení osy, uchopení dílu robotem R01_2360
- natočení osy, uchopení dílu robotem R01_2360
- přenos dílů robotem R01_2360 a založení
- současné svařování robotem R01_2360 a R02_2360

Stanice 2365 - robotické svařování

- uchopení dílu robotem R01_2365 a přenos ke svařovací stanici
- svařování na stacionární stanici robotem R01_2365
- odložení dílu robotem R01_2365
- otočení nástroje
- uchopení dílu robotem R01_2365 a přenos na další stůl

Stanice 2370 - robotické nanášení lepidla a svařování

- uchopení dílu z předchozího pracoviště robotem R02_2370 a přenos k lepící pistoli
- nanášení lepidla robotem R02_2370
- přenos dílu robotem R02_2370 na stůl
- současné svařování robotů R01_2370, R02_2370 a R03_2370
- uchopení dílu robotem R01_2375 a přenos ke svařovací stanici

Stanice 2375 - robotické svařování

- svařování na stacionární stanici robotem R01_2375
- frézování čepiček na svařovací stanici
- přenos dílu robotem R01_2375 na další stůl

3 BMS – 1 SK

Stanice 2380 - robotické svařování

- uchopení dílu z předchozího pracoviště robotem R01_2380 a přenos ke svařovací stanici
- svařování na stacionární stanici robotem R01_2380
- frézování čepiček na svařovací stanici
- přenos dílu robotem R01_2380 na stůl

Stanice 2381 - ruční zakládání dílu

- ruční zakládání dílu na dopravníkový přípravek

Stanice 2382 - robotické nanášení lepidla

- uchopení dílu robotem R01_2382 a přenos k lepící pistoli
- nanášení lepidla robotem R01_2382
- přenos dílu robotem R01_2382 na další stůl

Stanice 2385 - robotické svařování

- založení dílu robotem R01_2380
- založení dílu robotem R01_2382
- otočení nástroje o 180 stupňů
- současné svařování robotů R01_2382 a R01_2385
- uchopení dílu robotem R01_2386 a přenos ke svařovací stanici

Stanice 2386 - robotické svařování

- svařování na stacionární stanici robotem R01_2386
- frézování čepiček na svařovací stanici
- přenos dílu a založení robotem R01_2386 na další stůl

Stanice 2387 - robotické svařování

- uchopení dílu robotem R01_2387 a přenos ke svařovací stanici
- svařování na stacionární stanici robotem R01_2387
- frézování čepiček na svařovací stanici
- přenos dílu a založení robotem R01_2387 na další stůl

Stanice 2389 - robotické svařování

- uchopení dílu robotem R01_2389 a přenos ke svařovací stanici
- svařování na stacionární stanici robotem R01_2389
- frézování čepiček na svařovací stanici
- přenos dílu a založení robotem R01_2389 na další stůl

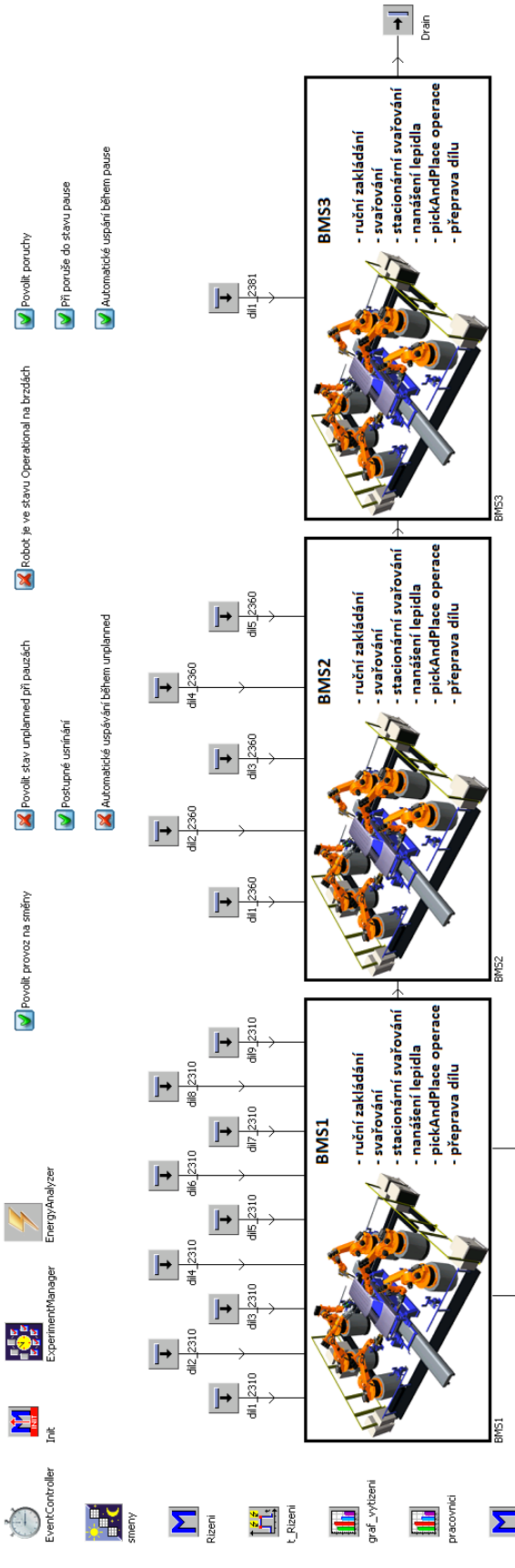
3 BMS – 2 SK

Stanice 2390 - manipulace se svařencem

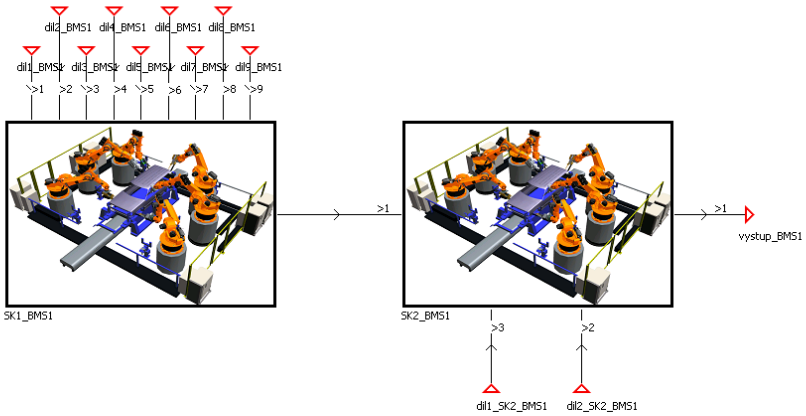
- uchopení dílu z předchozího pracoviště robotem R01_2390
- přenos hotového dílu na dopravník robotem R01_2390

9.2 Prostorové rozmístění linky

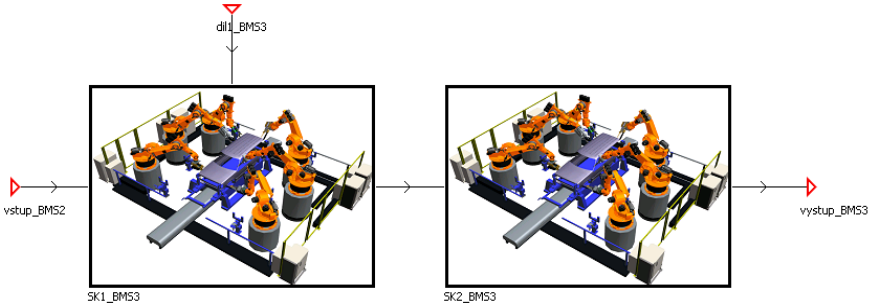
9.3 Plant Simulation model



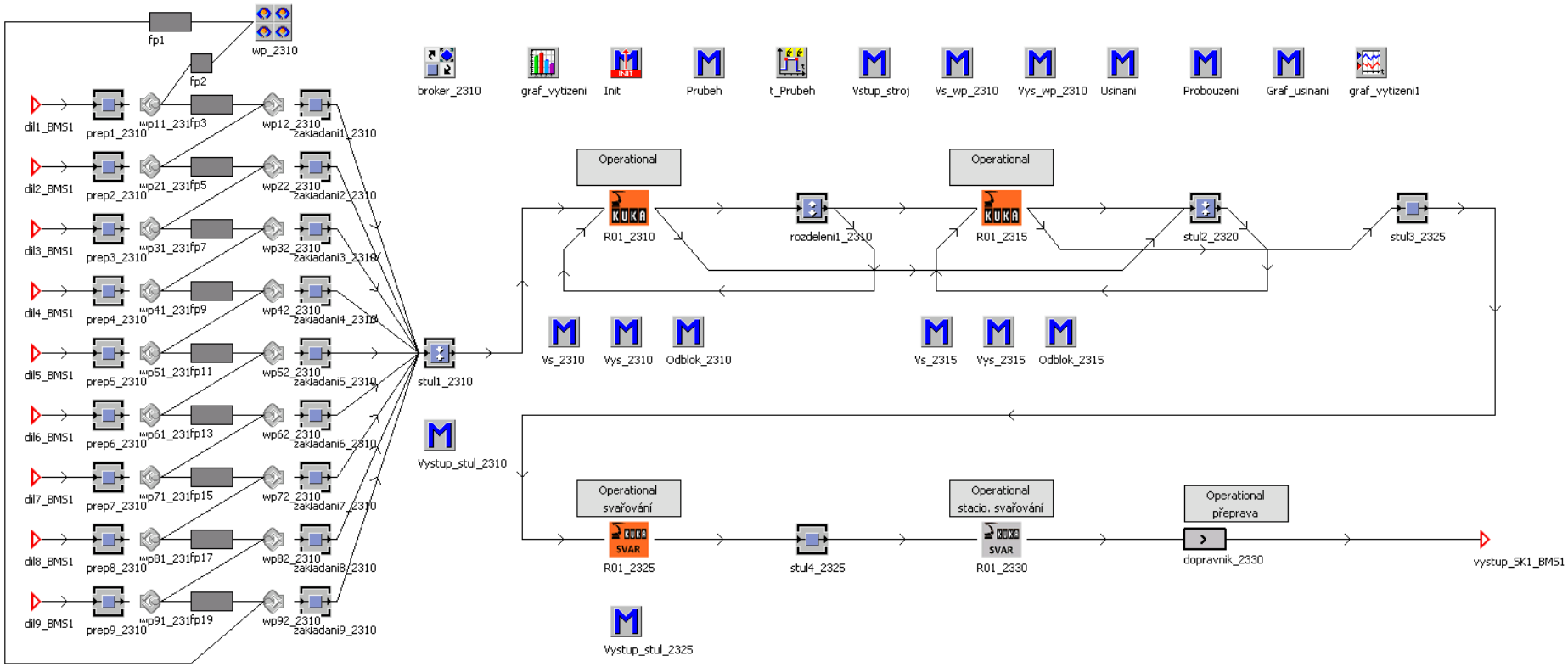
Linka


dil2_BMS1
dil4_BMS1
dil6_BMS1
dil8_BMS1
dil1_BMS1
dil3_BMS1
dil5_BMS1
dil7_BMS1
dil9_BMS1
>1
>2
>3
>4
>5
>6
>7
>8
>9
SK1_BMS1
>1
SK2_BMS1
>1
vystup_BMS1
>3
>2
dil1_SK2_BMS1
dil2_SK2_BMS1

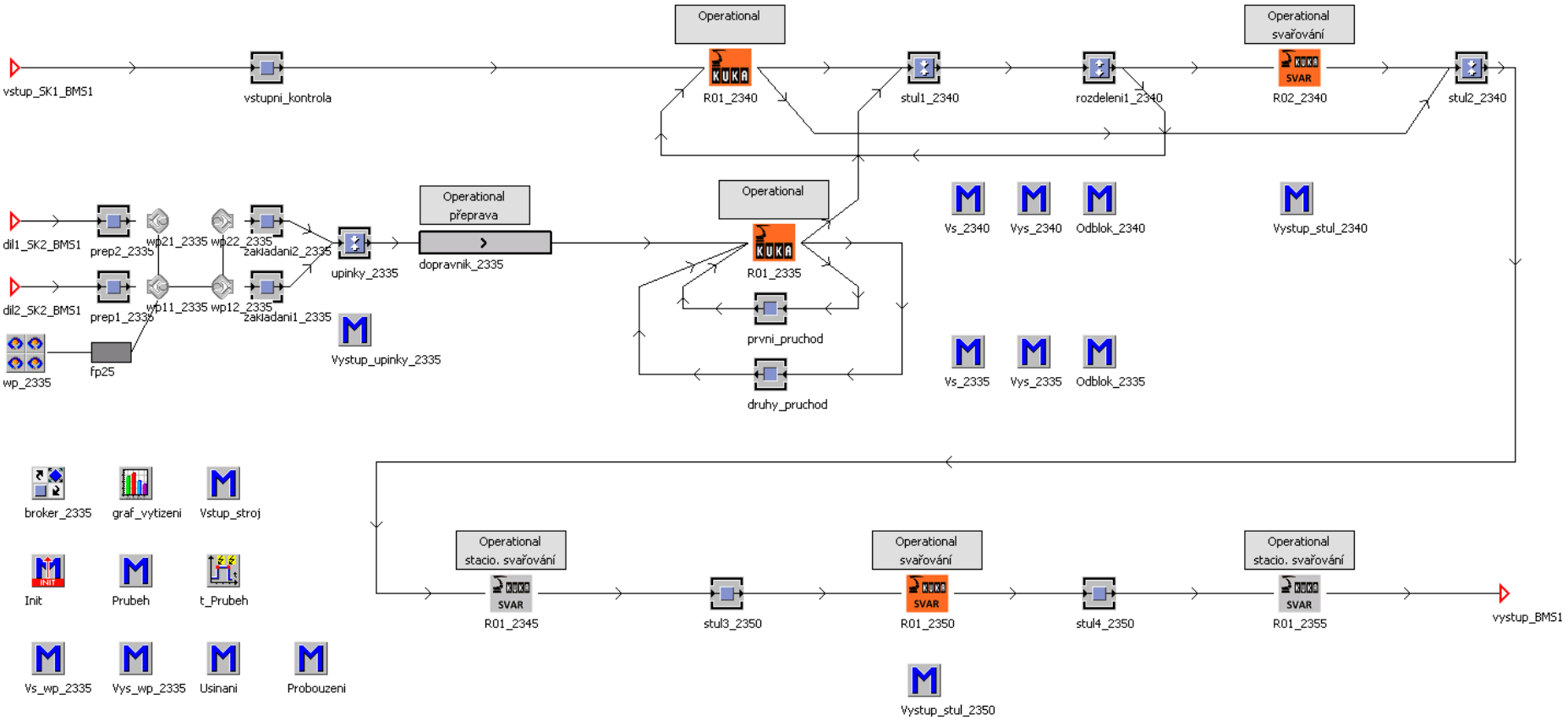
BMS1


dil1_BMS3
vstup_BMS2
SK1_BMS3
SK2_BMS3
vystup_BMS3

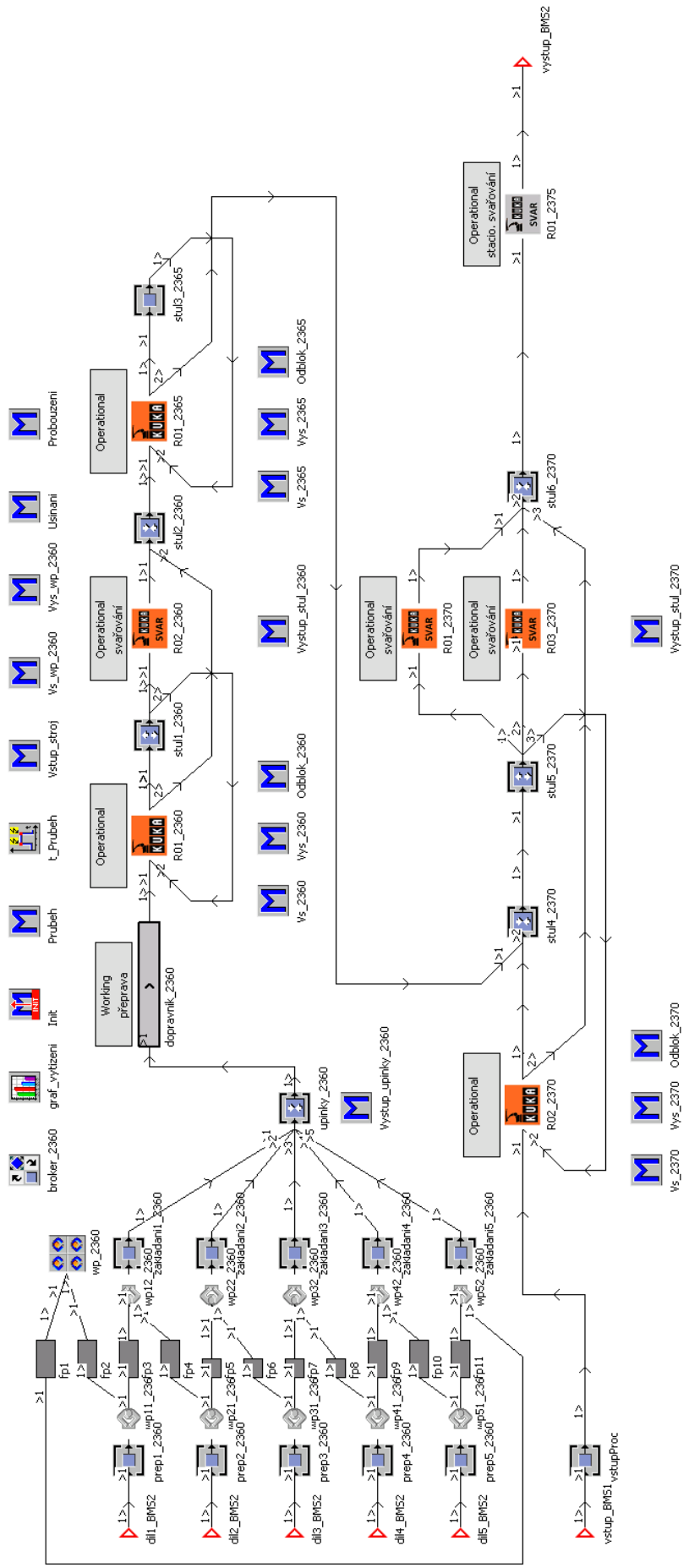
BMS3


fp1
wp_2310
fp2
broker_2310
graf_vytizeni
Init
Prubeh
t_Prubeh
Vstup_stroj
Vs_wp_2310
Vys_wp_2310
Usinani
Probouzeni
Graf_usinani
graf_vytizeni1
dil1_BMS1
prep1_2310
wp11_231fp3
wp12_2310
zakladani1_2310
dil2_BMS1
prep2_2310
wp21_231fp5
wp22_2310
zakladani2_2310
dil3_BMS1
prep3_2310
wp31_231fp7
wp32_2310
zakladani3_2310
dil4_BMS1
prep4_2310
wp41_231fp9
wp42_2310
zakladani4_2310
dil5_BMS1
prep5_2310
wp51_231fp11
wp52_2310
zakladani5_2310
dil6_BMS1
prep6_2310
wp61_231fp13
wp62_2310
zakladani6_2310
dil7_BMS1
prep7_2310
wp71_231fp15
wp72_2310
zakladani7_2310
dil8_BMS1
prep8_2310
wp81_231fp17
wp82_2310
zakladani8_2310
dil9_BMS1
prep9_2310
wp91_231fp19
wp92_2310
zakladani9_2310
stul1_2310
Operational
R01_2310
rozdeleni1_2310
Operational
R01_2315
stul2_2320
stul3_2325
Vs_2310
Vys_2310
Odblok_2310
Vs_2315
Vys_2315
Odblok_2315
Vystup_stul_2310
Operational svařování
SVAR
R01_2325
stul4_2325
Operational stacio. svařování
SVAR
R01_2330
Operational přeprava
dopravnik_2330
vystup_SK1_BMS1
Vystup_stul_2325

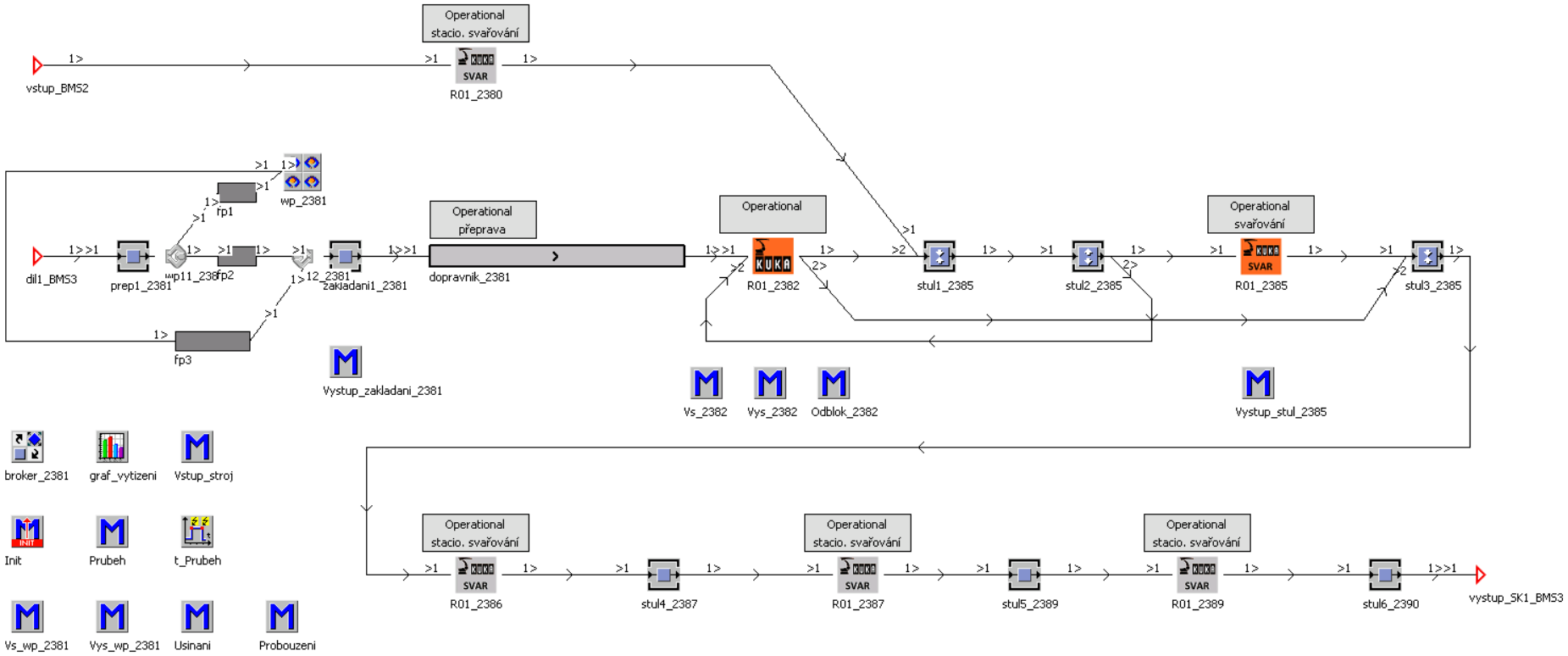
BMS1_SK1


Operational
Operational svařování
vstup_SK1_BMS1
vstupni_kontrola
R01_2340
stul1_2340
rozdeleni1_2340
R02_2340
stul2_2340
Operational přeprava
Operational
dil1_SK2_BMS1
prep2_2335
wp21_2335
wp22_2335
zakladani2_2335
upinky_2335
dopravnik_2335
R01_2335
Vs_2340
Vys_2340
Odblok_2340
Vystup_stul_2340
dil2_SK2_BMS1
prep1_2335
wp11_2335
wp12_2335
zakladani1_2335
Vystup_upinky_2335
prvni_pruchod
druhy_pruchod
Vs_2335
Vys_2335
Odblok_2335
wp_2335
fp25
broker_2335
graf_vytizeni
Vstup_stroj
Operational stacio. svařování
Operational svařování
Operational stacio. svařování
Init
Prubeh
t_Prubeh
R01_2345
stul3_2350
R01_2350
stul4_2350
R01_2355
vystup_BMS1
Vs_wp_2335
Vys_wp_2335
Usinani
Probouzeni
Vystup_stul_2350

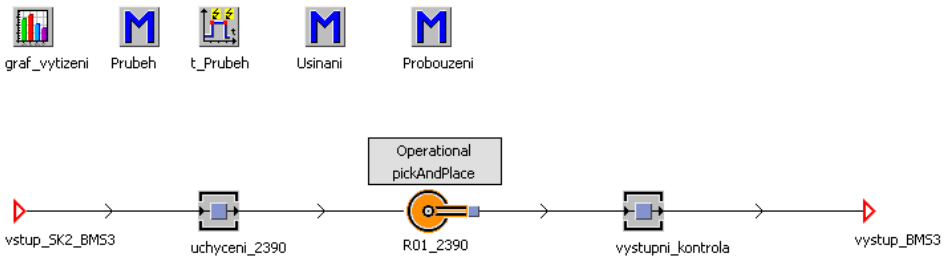
BMS1_SK2


broker_2360
graf_vytizeni
Init
Prubeh
t_Prubeh
Vstup_stroj
Vs_wp_2360
Vys_wp_2360
Usnani
Probouzeni
Working přeprava
dopravnik_2360
Operational
R01_2360
stul1_2360
Operational svařování
SVAR
R02_2360
stul2_2360
Operational
R01_2365
stul3_2365
Vs_2360
Vys_2360
Odblok_2360
Vystup_stul_2360
Vs_2365
Vys_2365
Odblok_2365
upinky_2360
Vystup_upinky_2360
wp_2360
wp12_2360 zakladani1_2360
wp22_2360 zakladani2_2360
wp32_2360 zakladani3_2360
wp42_2360 zakladani4_2360
wp52_2360 zakladani5_2360
fp1
fp2
wp11_2360
fp3
fp4
wp21_2360
fp5
fp6
wp31_2360
fp7
fp8
wp41_2360
fp9
fp10
wp51_2360
fp11
prep1_2360
prep2_2360
prep3_2360
prep4_2360
prep5_2360
dil1_BMS2
dil2_BMS2
dil3_BMS2
dil4_BMS2
dil5_BMS2
vstup_BMS1
vstupProc
Operational
R02_2370
stul4_2370
Operational svařování
SVAR
R01_2370
Operational svařování
SVAR
R03_2370
stul5_2370
Operational stacio. svařování
SVAR
R01_2375
stul6_2370
Vystup_stul_2370
Vs_2370
Vys_2370
Odblok_2370
vystup_BMS2

BMS2


Operational stacio. svařování
vstup_BMS2
R01_2380
wp_2381
fp1
Operational přeprava
Operational
Operational svařování
dil1_BMS3
prep1_2381
wp11_2381
fp2
12_2381
zakladani1_2381
dopravnik_2381
R01_2382
stul1_2385
stul2_2385
R01_2385
stul3_2385
fp3
Vystup_zakladani_2381
Vs_2382
Vys_2382
Odblok_2382
Vystup_stul_2385
broker_2381
graf_vytizeni
Vstup_stroj
Init
Prubeh
t_Prubeh
Vs_wp_2381
Vys_wp_2381
Usinani
Probouzeni
Operational stacio. svařování
Operational stacio. svařování
Operational stacio. svařování
R01_2386
stul4_2387
R01_2387
stul5_2389
R01_2389
stul6_2390
vystup_SK1_BMS3

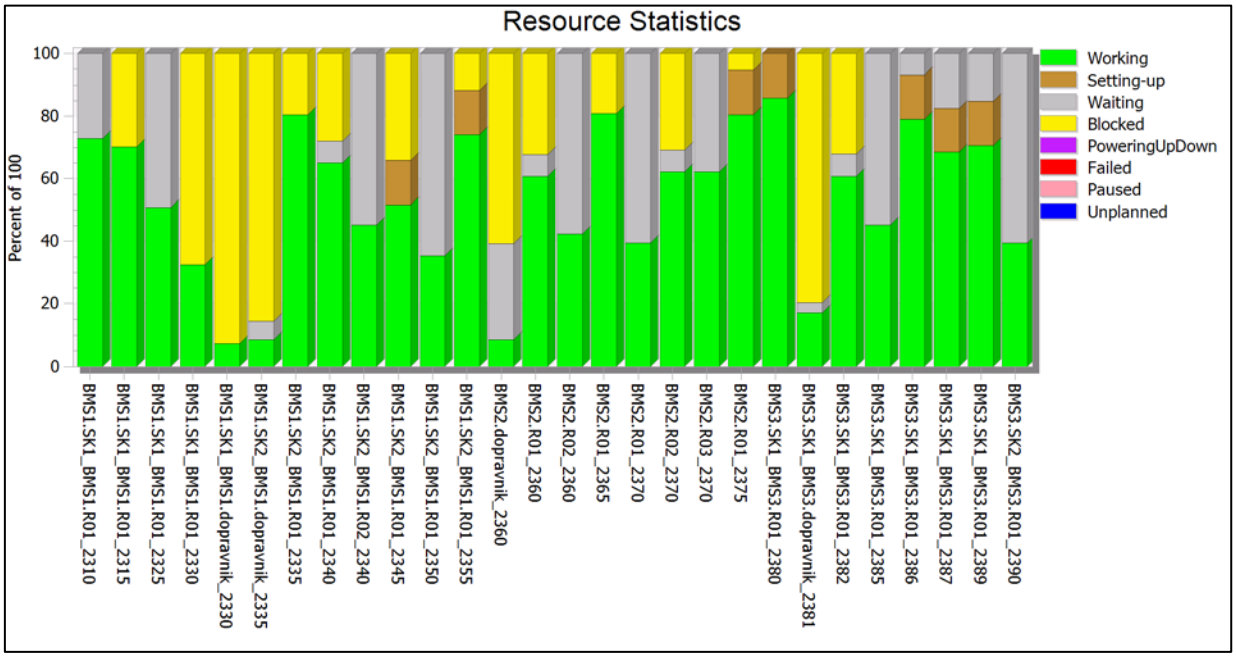
BMS3_SK1


graf_vytizeni
Prubeh
t_Prubeh
Usinani
Probouzeni
Operational
pickAndPlace
vstup_SK2_BMS3
uchyceni_2390
R01_2390
vystupni_kontrola
vystup_BMS3

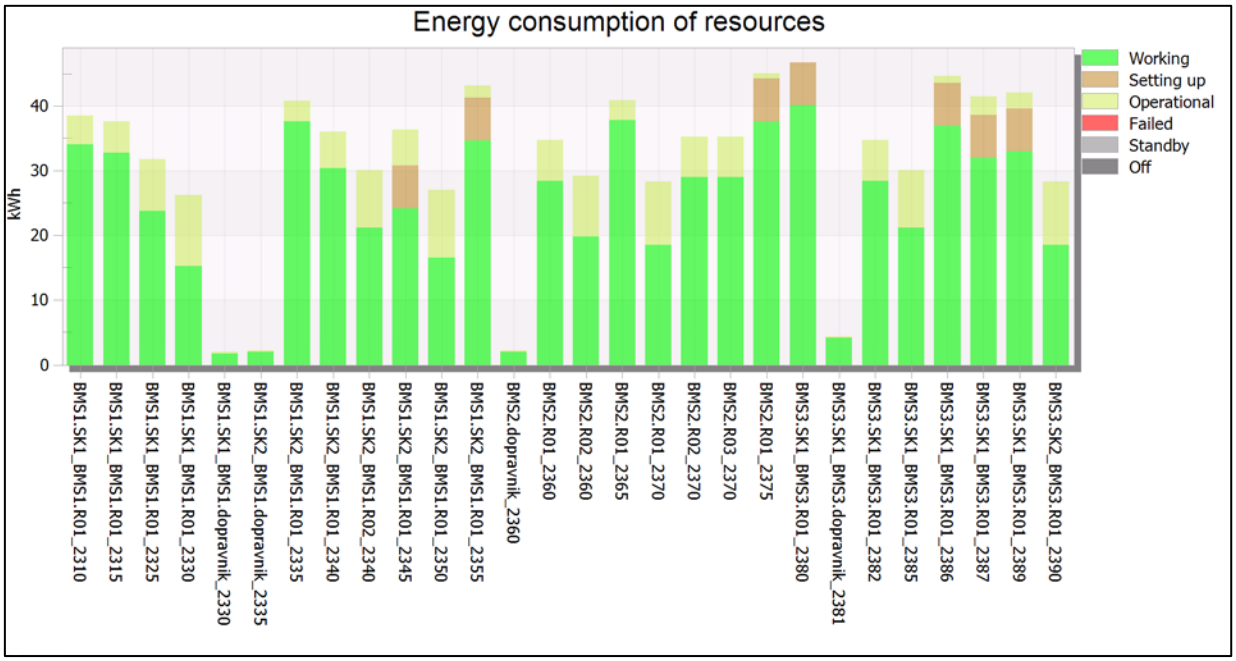
BMS3_SK2

9.4 Grafy

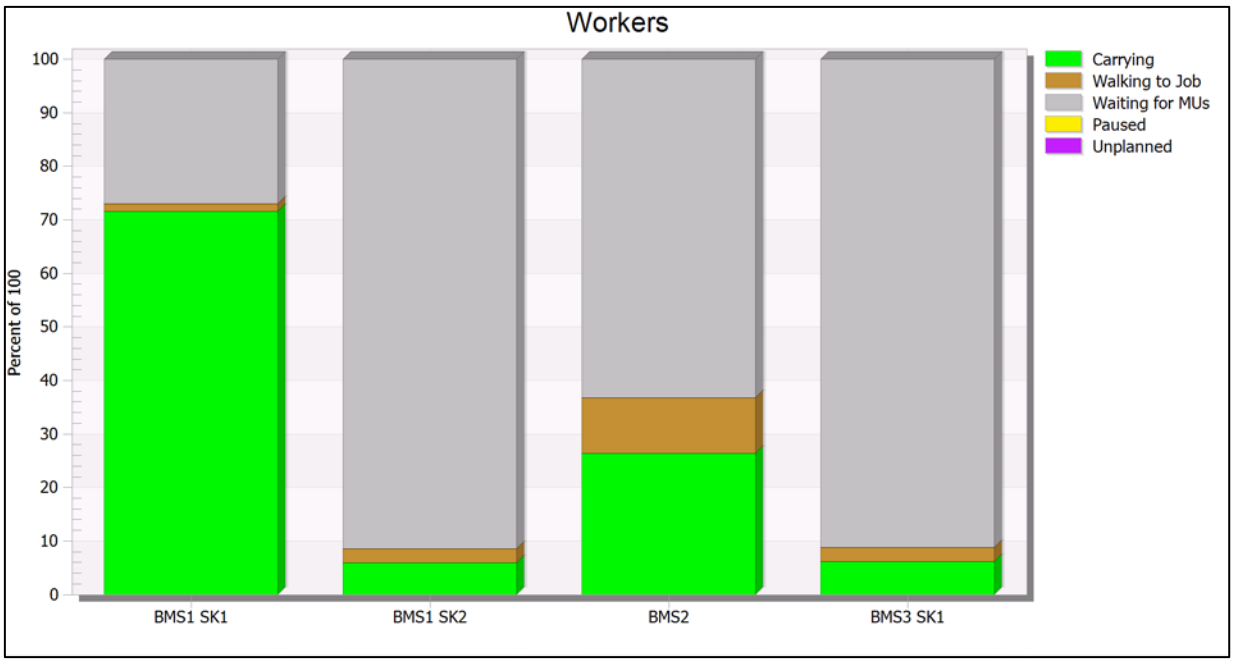
9.4.1 Základní analýza



Graf 9.1 Vytížení zdrojů (1 den)

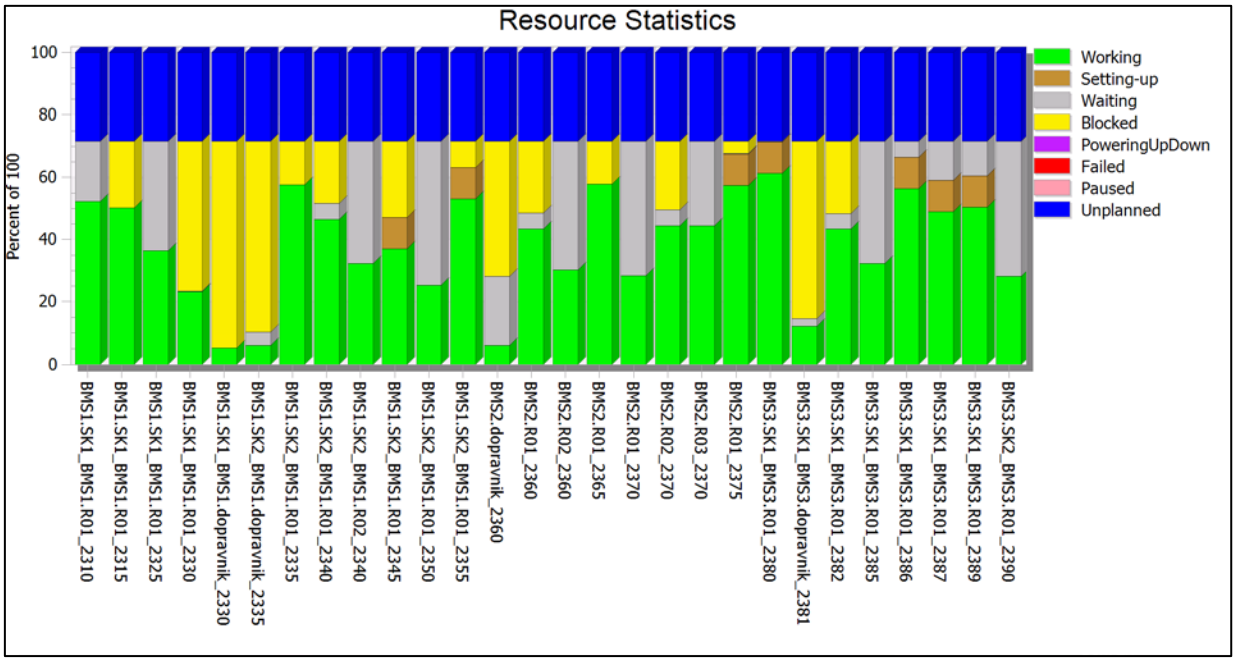


Graf 9.2 Spotřeba energie zdrojů (1 den)

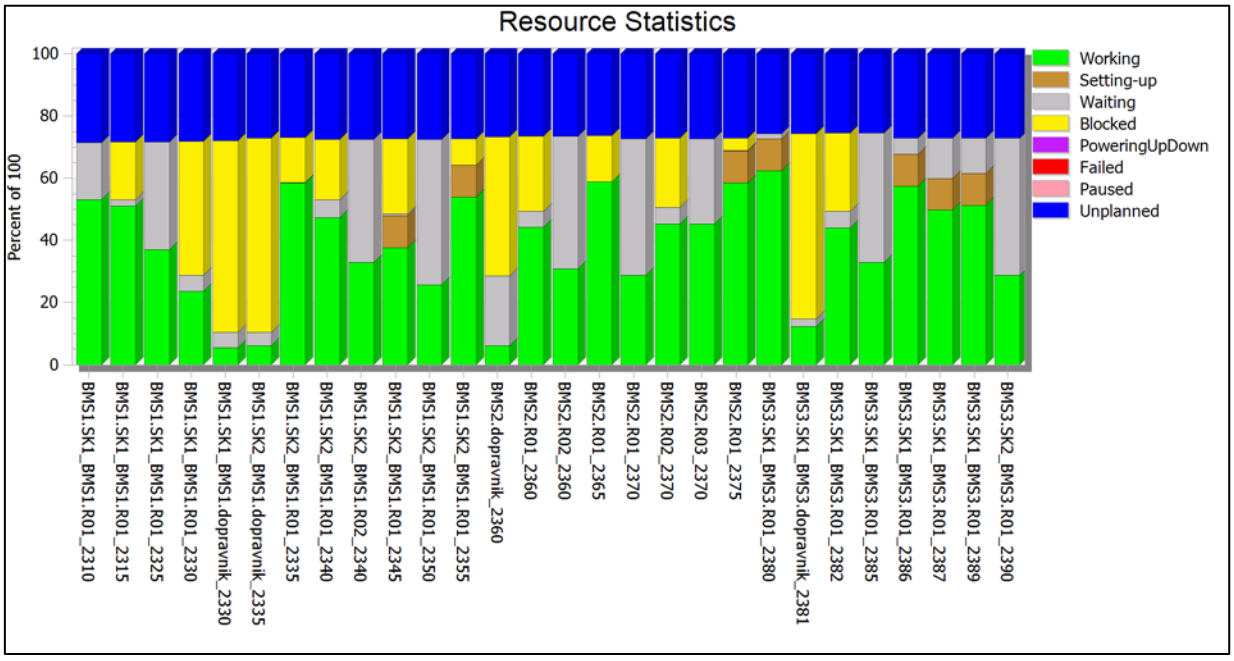


Graf 9.3 Vytížení pracovníků (1 den)

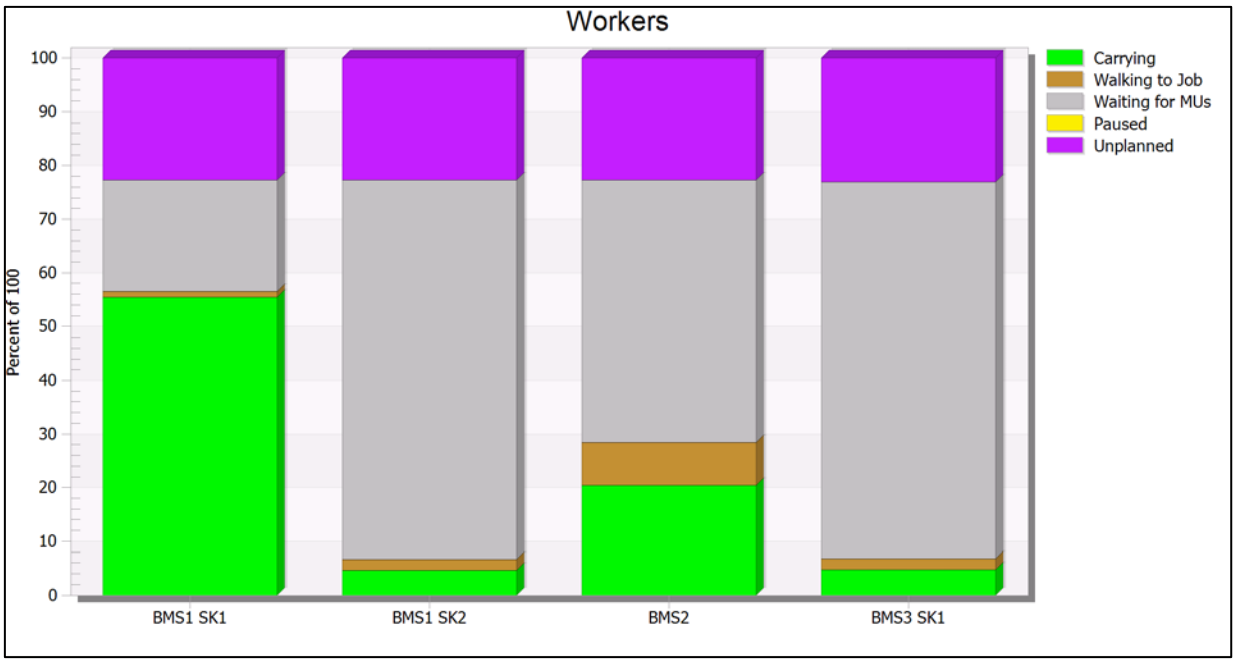
9.4.2 Směny



Graf 9.4 Vytížení zdrojů – Unplanned hromadně (7 dní)

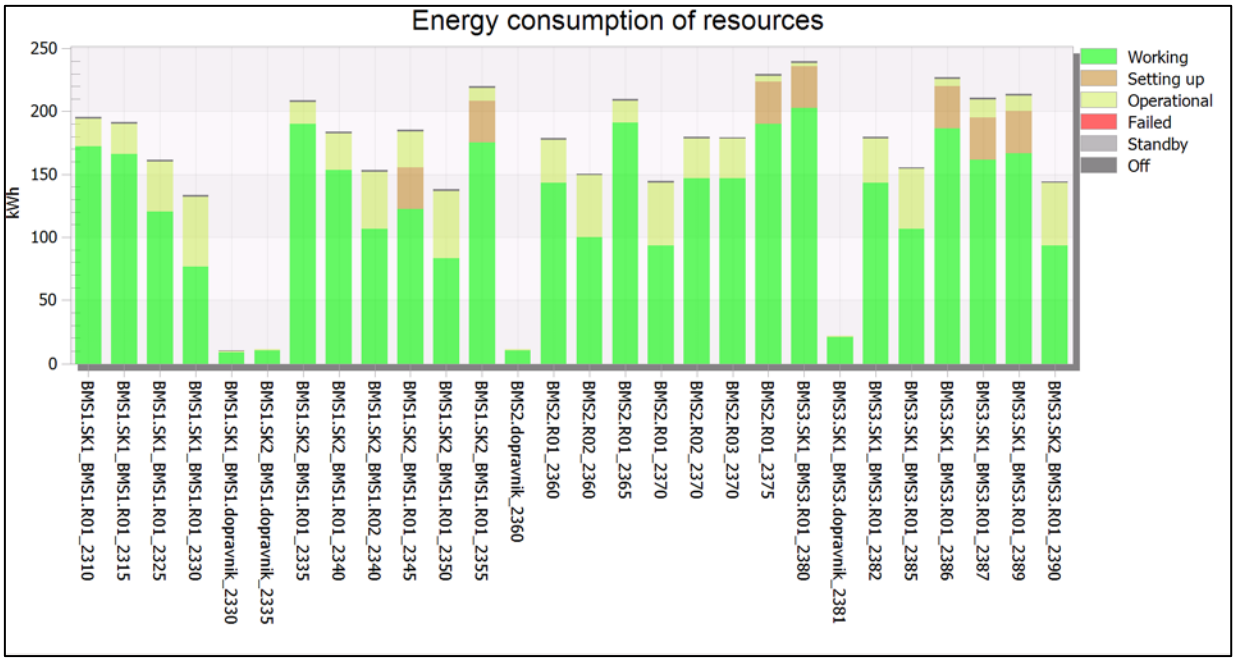


Graf 9.5 Vytížení zdrojů - Unplanned postupně (7 dní)



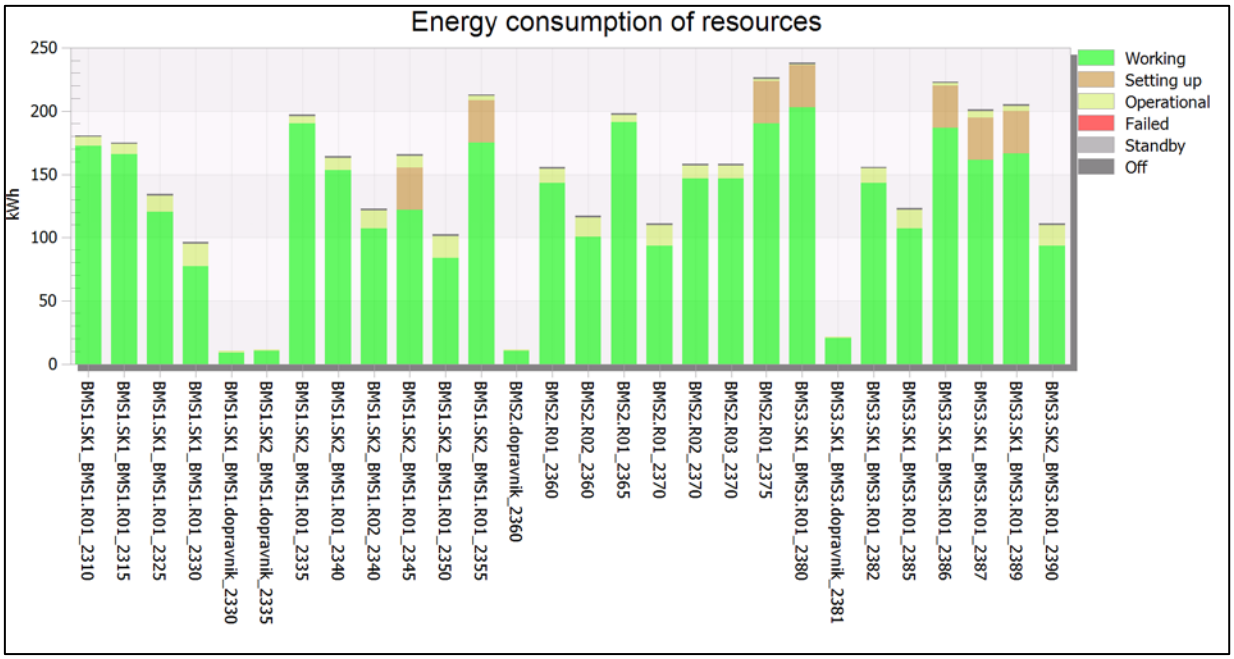
Graf 9.6 Vytížení pracovníků (7 dní)

9.4.3 Přechod do energeticky úsporného stavu



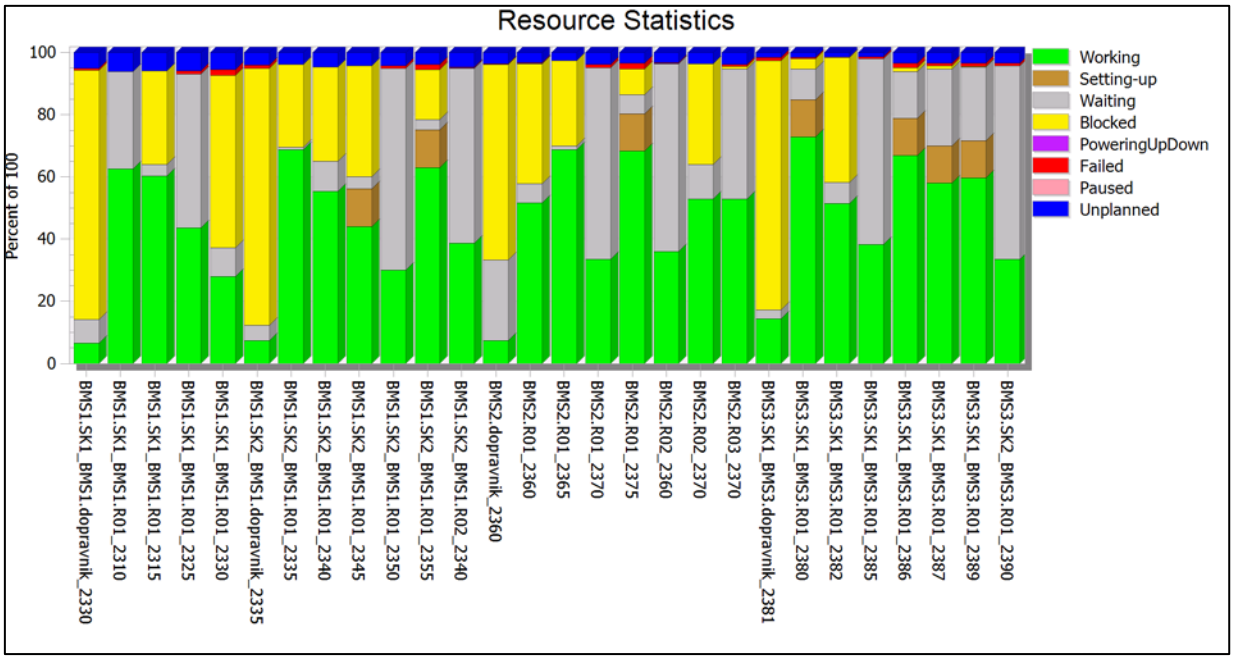
Graf 9.7 Spotřeba energie zdrojů - postupné uspávání (7 dní)

9.4.4 Přechod na brzdy robota

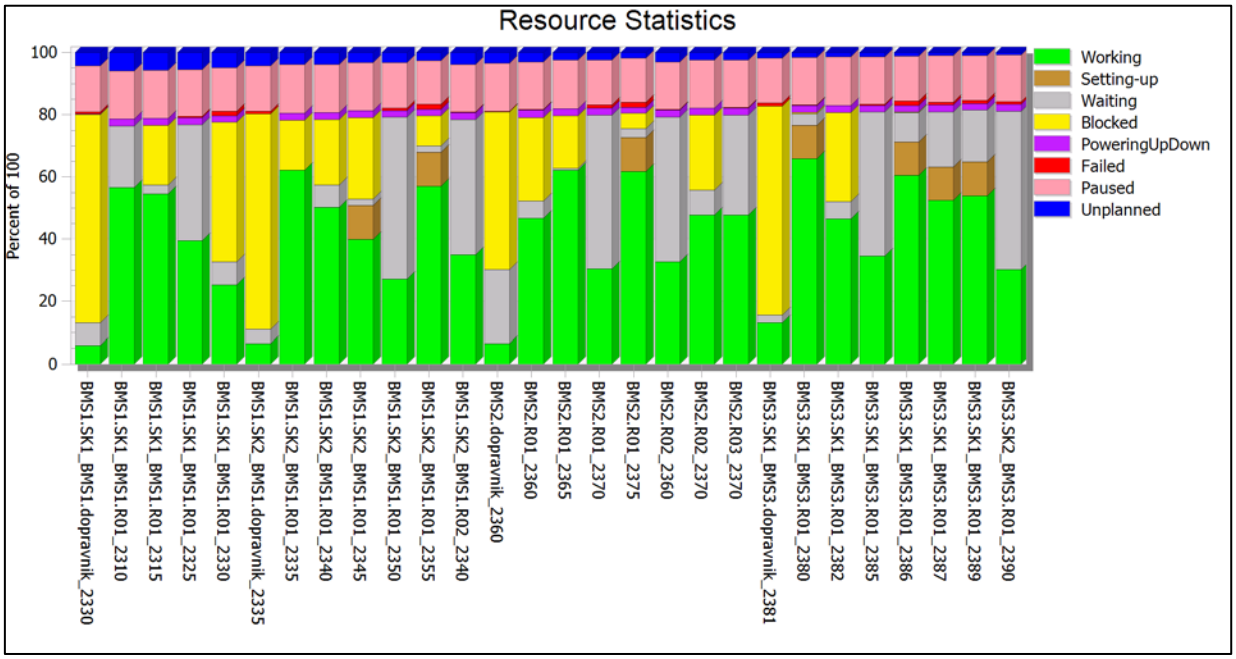


Graf 9.8 Spotřeba energie zdrojů - postupné uspávání + přechod na brzdy (7 dní)

9.4.5 Poruchy



Graf 9.9 Vytížení zdrojů - směny + poruchy bez usínání (1 den)



Graf 9.10 Vytížení zdrojů - směny + poruchy + usínání (1den)

9.5 Obsah CD

/PSim	projekt výrobní linky v Plant Simulationu
/prace	diplomová práce v elektronické podobě