Masarykova univerzita
Fakulta informatiky

Vizualizace modelů a biochemického prostoru v SBGN

Bakalárska práca

Michal Štefanič

Brno, jeseň 2018

MASARYKOVA UNIVERZITA
Fakulta informatiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Michal Štefanič

Program: Aplikovaná informatika

Obor: Bioinformatika

Garant oboru: prof. RNDr. Luboš Brim, CSc.

Vedoucí práce: RNDr. David Šafránek, Ph.D.

Katedra: Katedra strojového učení a zpracování dat

Název práce: Vizualizace modelů a biochemického prostoru v SBGN

Název práce anglicky: Visualisation of models and biochemical space in SBGN

Zadání: Cílem práce je nastudovat principy portálu e-cyanobacterium.org se zaměřením na biochemický prostor a jazyk BCSL a dále repozitář modelů. Klíčové objekty jsou pravidla (rules) v biochemickém prostoru a reakce (reactions) v dynamických modelech reprezentovaných dle standardu SBML. Student provede rešerši možností automatizované vizualizace v jazyku SBGN (mapování BCSL a matematických modelů do SBGN a relevantní knihovny pro vizualizaci).

V praktické části student navrhne způsob převodu BCSL a matematických modelů do SBGN a identifikuje část, pro niž realizuje implementaci spolupracující s portálem e-cyanobacterium.org prostřednictvím API.

Užitečné odkazy:

- http://sbgn.github.io/sbgn/
- https://www.e-cyanobacterium.org
- https://www.ebi.ac.uk/biomodels-main

Literatura:

KLEMENT, Matej, Tadeáš DĚD, David ŠAFRÁNEK, Jan ČERVENÝ, Stefan MUELLER a Ralf STEUER. *Biochemical Space: A Framework for Systemic Annotation of Biological Models*. In *Proceedings of the 5th International Workshop on Interactions between Computer Science and Biology (CS2Bio'14)*. Amsterdam: Elsevier, 2014. s. 31-44, 14 s. ISSN 1571-0661. doi:10.1016/j.entcs.2014.06.013.

TROJÁK, Matej, David ŠAFRÁNEK, Jakub HRABEC, Jakub ŠALAGOVIČ, Františka ROMANOVSKÁ a Jan ČERVENÝ. *E-Cyanobacterium.org: A Web-Based Platform for Systems Biology of Cyanobacteria*. In Ezio Bartocci et al.. *Computational Methods in Systems Biology. CMSB 2016.*. Heidelberg: Springer, 2016. s. 316-322, 7 s. ISBN 978-3-319-45176-3. doi:10.1007/978-3-319-45177-0_20.

Zadání bylo schváleno prostřednictvím IS MU.

Prohlášení autora školního díla

Jméno, příjmení a UČO studenta:_____

Beru na vědomí, že *Masarykova univerzita* může na základě zákona (§ 35 odst. 3 a 4 autorského zákona č. 121/2000 Sb.) užít mou kvalifikační práci nebo jiné mé školní dílo, které jsem jako autor vytvořil ke splnění svých studijních povinností vůči této vysoké škole, a to k výuce nebo k její vlastní vnitřní potřebě nikoli za účelem přímého nebo nepřímého obchodního nebo jiného hospodářského prospěchu.

Vlastní vnitřní potřebou *Masarykovy univerzity* se rozumí užití díla nejen v původní podobě, ale též ve zpracované nebo jinak změněné podobě zahrnující též takové užití mého díla touto vysokou školou, které spočívá v zadání mého školního díla k dalšímu zpracování jinému studentovi této vysoké školy (členovi téže akademické obce) za účelem vytvoření další kvalifikační práce nebo jiného školního díla, které bude odvozené od mého díla při uvedení mého autorství, názvu mého díla a pramene; a to vše v souladu s rozvojem vzdělanosti na *Masarykově* univerzitě a zájmem této veřejné vysoké školy navazovat na výsledky mé práce a mé školní dílo dále rozpracovávat v téže akademické obci.

Okolnosti hodné zvláštního zřetele z mé strany, například zájem o vlastní další rozpracování své kvalifikační práce na *Masarykově univerzitě* nebo jinde, jsem povinen sdělit této vysoké škole prostřednictvím studijního oddělení nejpozději při odevzdání kvalifikační práce.

V Brně dne_____ _____

podpis studenta

Vyhlásenie

Vyhlasujem, že táto bakalárska práca je mojím pôvodným autorským dielom, ktoré som vypracovala samostatne. Všetky zdroje, pramene a literatúru, ktoré som pri vypracovaní používala alebo z nich čerpala, v práci riadne citujem s uvedením úplného odkazu na príslušný zdroj.

Michal Štefanič

Vedúci práce: RNDr. David Šafránek Ph.D.

Poďakovanie

Ďakujem vedúcemu práce za vedenie práce a všetky cenné rady. Taktiež by som rád poďakoval členom laboratória SYBILA, za usmernenie a pomoc ako pri písaní práce, tak aj pri tvorbe aplikácie.

Zhrnutie

V tejto práci popisujeme návrh a implementáciu aplikácie, ktorá je schopná plne automaticky vizualizovať SBML reakcie a BCSL pravidlá biochemických modelov, v štandarde SBGN, ktorá bude následne použitá pre účely CMP. Práca taktiež zahrňuje návrh mapovania z jazyka BCSL do SBGN štandardu.

Abstract

In this thesis we describe design and implementation of the application, which is able to fully automaticly visualize SBML reactions and BCSL rules in SBGN standard, which will be used for purpose of the CMP. Thesis also include proposal of mapping from BCSL to SBGN standard.

Kľúčové slová

SBGN, BCSL, vizualizácia, e-cyanobacterium.org

Key words

SBGN, BCSL, visualization, e-cyanobacterium.org

Obsah

Úvod

Systémová biológia predstavuje holistický prístup, popisujúci komplexitu biologických systémov, ktorý začína pochopením, že biologický systém je niečo viac ako len súčet jeho menších častí. Živý organizmus ako aj jeho menšie komponenty, akými sú bunka či jadro, sú v systémovej biológii chápané ako systém. Dôležitú úlohu v týchto systémoch majú vzájomné interakcie medzi jeho časťami, a *emergentné vlastnosti*, čo sú vlastnosti systému, ktoré nemôžu byť pochopené len na základe súčtu vlastnosti jeho menších častí, ale predstavujú vlastnosti systému ako celku — komplexu. Nečakaná emergentná vlastnosť biologického systému môže byť výsledkom súhry príčin a následkov medzi jeho menšími komponentmi.

Pre štúdium a pochopenie biologických systémov a ich vlastností je tak nevyhnutné zhromažďovanie obrovského množstva dát. Informatika a počítačová technika sú rozhodujúce pre analýzu a modelovanie týchto dát. Pre zápis, ukladanie, analýzu a výmenu biochemických dát sa využíva množstvo rôznych formátov, pričom každý z nich sa snaží zachytávať študovaný organizmus s dôrazom na iné vlastnosti. Pre strojové spracovanie týchto dát sú vhodné písomné formáty, akým je napríklad System Biology Markup Language [1]. Naopak pre ľudí je omnoho jednoduchšie a rýchlejšie pochopiť jednotlivé súvislosti a vlastnosti organizmu z vizuálnej podoby týchto dát. Jedným z najpopulárnejších vizuálnych formátov v systémovej biológii je System Biology Graphical Notation [2].

Táto práca sa zameriava práve na vizualizáciu biochemických dát, pre účely Comprehensive Modelling Platform [3]. Cieľom práce bolo preskúmať možnosť prepojenia dvoch biologických formátov, medzi ktorými doteraz neexistuje žiadne známe mapovanie. Ide o formáty Biochemical Space language [4], ktorý formálne popisuje Biochemical space [5], využívaný v Comprehensive Modelling Platform na opis biologických systémov, a už spomínaný formát System Biology Graphical Notation.

Ďalším cieľom práce bolo navrhnúť a implementovať aplikáciu, ktorá príjme na vstup pravidlo v Biochemical Space language, alebo reakciu v System Biology Markup Language, spracuje dáta pomocou e-cyano API, vytvorí validný System Biology Graphical Notation súbor

a vygeneruje diagram, ktorý vráti na výstup vo formáte SVG alebo PNG.

V prípade úspešnej implementácie aplikácie a za predpokladu, že navrhnuté mapovanie bude dostatočne, vizuálne popisovať Biochemical Space language, bude aplikácia využívaná na Comprehensive Modelling Platform, kde bude predstavovať zlepšenie v oblasti vizualizácie biologických dát.

1 Základné pojmy

1.1 Model v systémovej biológii

Aby vedci nemuseli vykonávať všetky výpočty a experimenty v laboratóriách, čo by bolo ako nákladne, tak aj zdĺhavé, vytvorí sa na základe dát a informácií získaných či už sekvenovaním DNA alebo inými biochemickými postupmi *model* biologického systému, ktorý reprezentuje organizmus alebo jeho časť a zohráva kľúčovú úlohu v systémovej biológií [6, 7]. Vytváranie biochemických modelov je dôležité pre pochopenie vlastnosti organizmov, správanie sa systémov pri zmenách podmienok z rôznych príčin, či snaha predikovať tieto vlastnosti pomocou analýzy a simulácií.

Spomínané modely, rovnako ako všetky iné modely, ktoré si s týmto pojmom môžme spojiť (model auta, mesta, slnečnej sústavy), sú abstrakciou reality. Zameriavajú sa na jeden konkrétny aspekt študovaného objektu a často zanedbávajú pohľad na jeho ostatné vlastnosti.

V technickom priemysle sa často pracuje s matematickým modelom, ktorý sa používa na analýzu a dizajn rôznych objektov. Každý takýto objekt ma presne určenú svoju funkciu, vďaka čomu je možné ľahko vykonávať rôzne výkonnostné testy, a posúdiť tak efektivitu a robustnosť týchto funkcií. Napriek tomu, že biologické systémy nie sú žiadne technické objekty a sú prirodzenou súčasťou prírody, existujú kvôli tomu, že v nej zohrávajú určitú funkciu. Tieto biologické funkcie živých organizmov, vieme podobne ako v technickom priemysle, za pomoci rôznych postupov, zapísať a študovať. Tato analógia s technickým priemyslom ma však svoje hranice. Napríklad prirodzený výber nevieme simulovať a zatiaľ je to čisto záležitosť prírody, aj keď umelá inteligencia, či virtuálna evolúcia sa už pokúšajú porozumieť evolučným procesom prostredníctvom počítačovej simulácie jednoduchých (umelých) foriem života.

Biologické modely môžu byť zapísané pomocou grafov alebo súborom matematických či chemických rovníc. Najrozšírenejším štandardným formátom, slúžiacim na ukladanie a výmenu modelov v systémovej biológii, je System Biology Markup Language (SBML) [1], ktorý je bližšie popísaný v ďalšej časti tejto kapitoly. Veľké množstvo mode-

lov zapísaných práve vo formáte SBML je možné nájsť na platforme BioModels. Ďalšími populárnymi štandardmi sú BioPAX a CellML.

V prípade, že máme vytvorený a uložený model, v niektorom populárnom formáte, je možné pomocou nástrojov určených na analýzu a simulácie biologických modelov, jeho*spustenie* s určitými parametrami a počiatočnými podmienkami, ktoré sú v konkrétnom zápise modelu zadefinované. Simulácia modelu sa snaží čo najpresnejšie priblížiť reálne správanie systému a pomocou experimentov a zmien parametrov modelu, následne predikovať správanie a vytvárať rôzne hypotézy.

1.2 Modelovanie založené na pravidlách

Hlavnou zložkou každého biologického procesu sú reakcie, látky, ktorých zmenu reakcia popisuje a parametre popisujúce rôzne vedľajšie faktory, akým je napríklad vplyv prostredia. Látka predstavuje elementárny objekt reakcie, ktorý môže zohrávať buď úlohu reaktantu — látka vstupujúca do reakcie, produktu — látka, ktorá vzniká ako výsledok reakcie alebo modifikátor — látka, ktorá reakciu ovplyvňuje, ale jej štruktúra sa nemení. Samotná reakcia popisuje postup, ako sa z reaktantov, za pomoci modifikátorov, stávajú produkty.

V modelovaní biologických systémov sa preto bežne používa modelovanie založené na reakciách. Všeobecný problém s týmto prístupom je, že vyžaduje popis všetkých látok a reakcií, vyskytujúcich sa v systéme. Toto množstvo je však často príliš veľké, aby sa s takým modelom dalo efektívne pracovať. Riešením je práve modelovanie založené na pravidlách. Ide o rozšírenie prístupu založeného na reakciách, kde sa namiesto látok, pracujeme s typmi. Reakcie popisujúce zmeny látok sú nahradené pravidlami, ktoré definujú zmeny jednotlivých typov na iné. Model vytvorený týmto prístupom je potom zvyčajne omnoho stručnejší.

Model založený na pravidlách, je definovaný množinou pravidiel a počiatočným náhodným rozložením vzájomne na seba pôsobiacich objektov. Keďže rozloženie je náhodné, nemôžeme objektom prideliť žiadne poradie a z biologického hľadiska, je toto rozloženie čo najbližšie k realite, zatiaľ čo sa snaží zachovať stručnosť. Pravidlá popisujú správanie pre skupinu objektov. Pravidlo môžme chápať ako

abstraktnú chemickú reakciu, kde hlavný rozdiel oproti reakcii je, že reakcia pracuje iba s konkrétnymi objektmi, zatiaľ čo pravidlo pracuje so skupinou interagujúcich objektov. Môžme teda tvrdiť, že reakcia je špeciálny prípad pravidla, kde typ prestavuje práve jeden objekt.

1.3 Comprehensive Modelling Platform

Po krátkom úvode do systémovej biológie a modelovaniu biologických systémov sa dostávame k projektu *Comprehensive Modelling Platform* (CMP) [3]. CMP je projekt, ktorého cieľom je vytvoriť webovú platformu, na verejné zdieľanie, anotáciu, analýzu a vizualizáciu biologických modelov a experimentov, súvisiacich s určitým biologickým organizmom. Konkrétnou inštanciu CMP je už existujúci portál *e-cyanobacterium.org* [8] (ďalej e-cyano), kde sa nachádzajú modely a experimenty súvisiace so *sinicami* a vďaka ktorému myšlienka CMP vznikla. Portál bol vytvorený a je spravovaný na *Fakulte informatiky Masarykovej univerzity* tímom laboratória SYBILA. CMP používateľom umožní nahrať si modely a experimenty konkrétneho organizmu, následne vykonávať rôzne simulácie, analýzy, či vizualizovať a exportovať modely v rôznych formátoch.

Modely na platforme sú reprezentované rôznym typom abstrakcií, akými sú napríklad biochemické reakčné siete alebo diferenciálne rovnice. Platforma je jedinečná v poskytovaní stručného mapovania matematických modelov do formálneho biochemického popisu. Hlavným cieľom platformy je spojiť svet biologických poznatkov s výhodami matematického popisu dynamických procesov. Samotná platforma je tvorená viacerými modulmi. Jedným z hlavných je *Biochemical space*, s ktorým sú prepojené všetky ostatné moduly.

1.3.1 Biochemical space

Biochemical space (BCS) [5] predstavuje základ platformy a poskytuje formálny biochemický popis a anotáciu biologického systému. Je založený na hierarchii vybraných biologických procesov a objektov. Veľmi zjednodušene, BCS slúži na popis biológie. Cieľom je poskytnúť biologický základ pre matematické modely. Na dosiahnutie tohto cieľa, BCS

využíva databázu biochemických objektov, čo umožňuje popisovať konkrétne objekty so všetkými ich vlastnosťami a atribútmi.

Každý objekt by mal byť správne anotovaný, doplnený podrobnými informáciami a odkazmi na príslušné interné a externé zdroje a databázy. Objekty však na popis modelov nestačia, pretože modely vyžadujú taktiež hierarchiu procesov popisujúcu interakcie medzi objektmi. Preto spolu s priestorovou hierarchiou objektov je vytvorená aj hierarchia procesov, prostredníctvom interaktívnych schém. Táto hierarchia nám umožňuje priblížiť sa k jednotlivým pravidlám, ktoré predstavujú atomické operácie, odohrávajúce sa v biologickom systéme.

BCS predstavuje ľudský čitateľný formát, ktorý možno ľahko editovať vo vyhradenom editore a vizualizovať v rámci platformy. BCS, ako už bolo spomenuté, je definované dvoma časťami — množinou objektov a množinou pravidiel. Pre zlepšenie použiteľnosti a validácie BCS je definovaný *Biochemical Space Language* (BCSL) [4], ktorý formálne definuje jednotlivé objekty, pravidlá a samotný model. BCSL je bližšie popísaný v ďalšej časti tejto kapitoly.

V BCS sa kladie veľký dôraz na správnu anotáciu jednotlivých objektov. Každý objekt pozostáva z atribútov, ktoré si popíšeme na príklade hydrogén-uhličitanu:

- **ID entity:** *HCO3*
- **stavy:** *{-, +}*
- **lokácia:** *cyt, liq*
- **názov:** *hydrogén-uhličitan*
- **klasifikácia:** *malá molekula*
- **popis:** *Zohráva dôležitú rolu v mechanizme koncentrácie uhlíka (CCM)*
- **odkazy:** *CHEBI::17544*
- **organizmus:** *Synechococcus elongatus PCC 7942*

Objekt v BCS môže definovať rôzne biochemické látky. Od najmenších atomických objektov (*HCO3*), ktoré sa môžu vyskytovať v rôznych

stavoch, cez väčšie štruktúrne objekty (*foto-systém*) zložené z atomických objektov, až po komplexy (*proteín*) tvorené ako atomickými či štruktúrnymi objektami, tak aj ďalšími vnorenými komplexmi. Špeciálnou entitou sú takzvané kompartmenty (*cytoplazma*), ktoré reprezentujú priestor, kde sa zvyšné entity nachádzajú.

Pravidlá sú definované špeciálnymi rovnicami obsahujúcimi dodatočné informácie. Strany rovnice, na ktorých sa nachádzajú jednotlivé produkty a reaktanty, rovnako ako pri reakciách, sú oddelené znamienkom +. Rozlišuje sa typ pravidla, ktoré môže byť buď nevratné (*ireverzibilné*) alebo vratné (*reverzibilné*). Každý jeden objekt v pravidle ma priradený svoj kompartment, čo hraje dôležitú úlohu hlavne pri pravidlách, kde sa vyskytuje viacero rôznych kompartmentov. Pred každým objektom sa môže nachádzať koeficient udávajúci jeho kvantitu. Niektoré pravidlá môžu obsahovať modifikátor — napríklad enzymatické reakcie. Keďže BCS dovoľuje teoreticky *nekonečné* zanorovanie sa, môže dochádzať k vzniku pomerne zložitých a rozsiahlych modelov.
Ďalšími hlavnými modulmi portálu sú *Model repository* a *Experiments repository*.

1.3.2 Model repository

Model repository je kolekcia matematických modelov, popisujúcich konkrétne častí biologických procesov. Každý model je tvorený súborom matematických rovníc, generovaných z modelu reakčnej siete. Každý objekt modelu je prepojený a mapovaný do BCS. To znamená, že každý objekt modelu je mapovaný na konkrétny objekt v BCS, a každá reakcia je mapovaná na konkrétne pravidlo z BCS. Model tak obsahuje kompletné informácie obohatené o biologické notácie, pre všetky látky a reakcie. Model tiež obsahuje počiatočné parametre, ktoré umožňujú jeho simuláciu. Model je možné exportovať do SBML.

1.3.3 Experiments repository

Experiments repository je modul slúžiaci pre ukladanie a prezentáciu takzvaných *time-series* dát z laboratórnych experimentov. Každý experiment je podložený detailným popisom a dôkladnou anotáciou. Jed-

notlivé premenné a zložky experimentu môžu byť prepojené s BCS. Výsledky experimentov je možné vizualizovať pomocou grafov.

1.4 Formáty modelov v systémovej biológii

V tejto časti práce sú opísané tri formáty systémovej biológie, ktorými môžu byť modely definované. Nejde o jediné formáty, ktoré existujú, ale ide o formáty relevantné pre koncept tejto práce. Ide o už spomínaný SBML, *System Biology Graphical Notation* (SBGN) [2] a BCSL.

1.4.1 SBML

SBML [1] definuje modely biochemických objektov viazaných a modifikovaných procesmi — biochemickými reakciami. V SBML sa jednotlivé biochemické objekty nazývajú entity. SBML model je tvorený väčším počtom rozdielnych komponent: reaktanty, produkty, reakcie, reakčné rýchlosti, ... Aby bolo model možné analyzovať, či simulovať, musia byť explicitne uvedené ďalšie komponenty vrátane kompartmentov, v ktorých sa nachádzajú objekty v rôznych množstvách.

V SBML sa môžu nachádzať nasledujúce komponenty:

- **definícia funkcie** — pomenovanie matematickej funkcie, ktorá môže byť modelom používaná
- **definícia jednotky** — definovanie novej jednotky merania, alebo zmena definície základnej jednotky SBML
- **typ kompartmentu** — typ prostredia kde sa entity reakcie — chemické látky môžu nachádzať
- **typ entity** — charakterizuje typ entity, ako napríklad ión vápnika, molekulu glukózy, či ATP, ktorá prináleží určitej reakcii
- **kompartment** — množina kompartmentov rôznych typov a konečnej veľkosti, ktoré definujú kde sa jednotlivé entity SBML modelu môžu vyskytovať. V modely sa môže nachádzať viacero kompartmentov rovnakého typu. Každá entita modelu, musí byť lokalizovaná aspoň v jednom z týchto kompartmentov.

- **entita** — množina entít rovnakého typu, ktoré ktoré sú lokalizované v určitom kompartmente
- **parameter** — vyjadruje určité množstvo, pod vymysleným symbolickým názvom. Parameter je použitý vo všeobecne, aby nerozlišoval či ide o konštantu alebo o premennú reakcie. Parameter definovaný na najvyššej vrstve je chápaný ako globálna premenná, ale parameter je taktiež možné si zadefinovať aj pre jednu konkrétnu reakciu, teda ako lokálnu premennú.
- **iniciálne nastavenia** — matematický výraz vyjadrujúci počiatočné podmienky modelu
- **pravidlo** — matematický výraz priradený k súboru rovníc, vytvorených na základe reakcií definovaných v modely. Pravidlá definujú ako počítať hodnotu určitej premennej z iných premenných a taktiež je nimi možné definovať mieru zmeny premennej. Spolu s rovnicami reakčnej rýchlosti sa s pravidlami dá určiť správanie modelu v reálnom čase.
- **obmedzenie** — zakazujú neplatné stavy a podmienky modelu počas celej simulácie. Sú definované matematickými výrazmi, ktoré vypočítajú pravé alebo falošné hodnoty z modelových premenných, parametrov a konštánt.
- **reakcia** — výraz popisujúci určitý proces transformácie, prenosu alebo väzby, ktorý môže meniť množstvo jedného alebo viacerých entít. Reakcia popisuje ako sa určité entity (reaktanty) transformujú na iné entity (produkty). Reakcie majú priradené výrazy, (ľubovoľné matematické funkcie) definujúce kinetickú rýchlosť, ktorá určuje ako rýchlo sa uskutočňujú.
- **udalosť** — popisuje okamžitú nespojitú zmenu množiny premenných akéhokoľvek typu (množstvo, veľkosť kompartmentu, hodnotu parametra, ...), za predpokladu, že je splnená spúšťacia podmienka tejto udalosti

Okrem vyššie spomenutých komponent, ďalšou dôležitou vlastnosťou SBML je, že každý objekt môže obsahovať strojovo čitateľné anotácie, ktoré vyjadrujú vzťahy medzi objektmi z modelu a objektmi

z externých zdrojov, napríklad biochemické databázy. S dôkladnými anotáciami, model nie je len obyčajným matematickým zápisom, ale stáva sa sémanticky obohateným nástrojom na komunikáciu a šírenie vedeckých poznatkov.

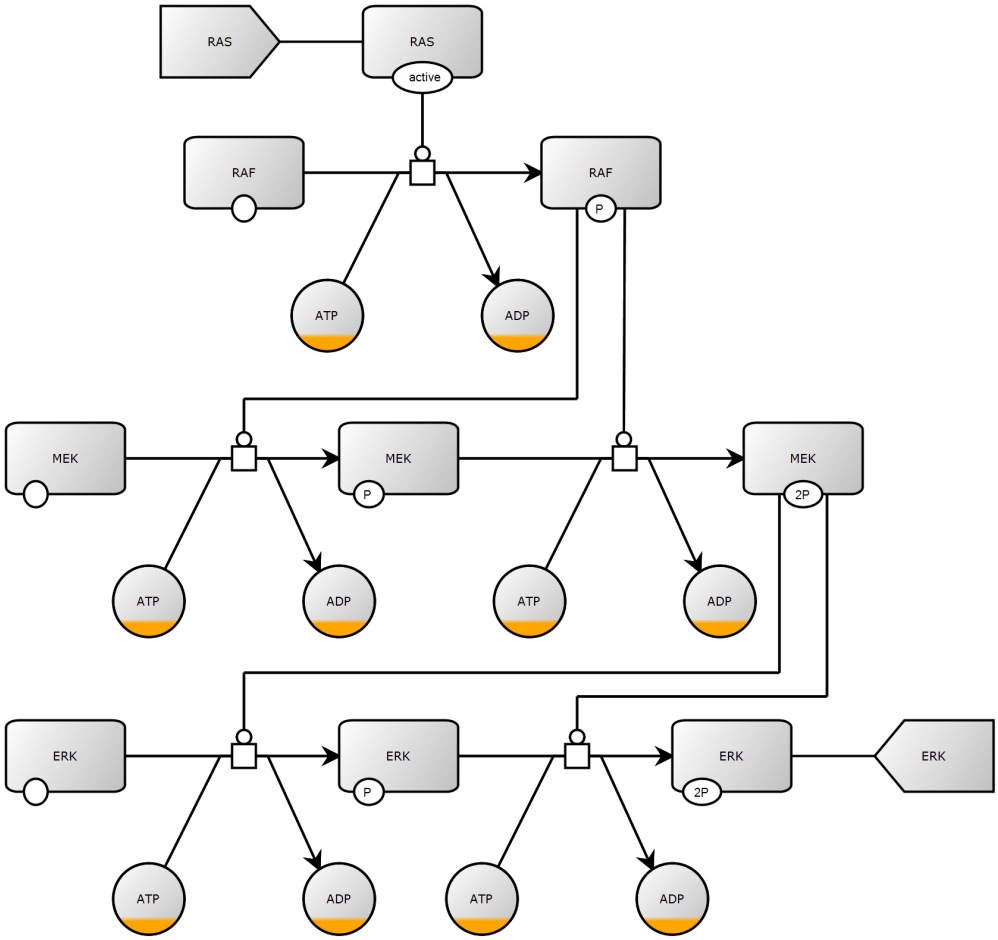
1.4.2 SBGN

SBGN je grafický štandard, slúžiaci na efektívne ukladanie, výmenu a vizualizáciu dát, obsahujúcich informácie o biologických sietiach ako napríklad signalizačné dráhy, metabolické siete či génové regulačné siete. SBGN niekoľko rokov vytváralo spoločenstvo biochemikov, modelárov a počítačových vedcov [2]. Dáta sú uložené v súboroch v rozšíriteľnom značkovacom jazyku (XML), kde obsahujú informácie o jednotlivých komponentách a ich priestorovom rozložení. Každá komponenta ma svoje súradnice, výšku a šírku. SBGN pozostáva z troch jazykov, ktoré zachytávajú rôzne pohľady na biologické systémy:

- ***Process description language*** **(PD)** je jazyk, ktorý zobrazuje časové priebehy biochemických interakcií v sieti. Môže byť použitý na zobrazenie všetkých molekulárnych interakcií, ku ktorým dochádza v sieti medzi biochemickými objektmi.
- ***Entity relationship language*** **(ER)** tento jazyk zobrazuje všetky vzťahy, v ktorých sa objekt môže vyskytovať, bez ohľadu na čas. Vzťah je možné chápať ako pravidlo popisujúce vplyv objektov na iné vzťahy.
- ***Activity flow language*** **(AF)** je posledný jazyk a slúži na zobrazovanie toku informácií medzi objektmi v sieti. Neobsahuje informácie o zmenách stavov jednotlivých objektov a je vhodný na zobrazenie odchýliek či už genetických alebo environmentálnych v prírode.

Každý z jazykov je definovaný komplexnou sadov symbolov s presnou sémantikou a podrobnými syntaktickými pravidlami, týkajúcimi sa konštrukcie a interpretácie grafických štruktúr. Pomocou týchto troch jazykov je možné graficky zachytiť procesy prebiehajúce v biologických sieťach. Pre koncept práce je dôležitý hlavne prvý z menovaných jazykov — PD, ktorý zároveň patrí medzi najviac používaný.

1.4.3 SBGN-PD



Obr. 1.1: Príklad SBGN diagramu

Úlohou PD je zobraziť ako odlišné objekty v systéme prechádzajú z jednej formy do druhej. Autorom jazyka bolo od začiatku vývoja jasné, že je nemožné vytvoriť kompletnú a presnú formu jazyka hneď po prvýkrát. Preto sa rozhodli rozdeliť vývoj jazyka do viacerých úrovní. Pod konkrétnou úrovňou jazyka SBGN sa rozumie sada funkcií, ktoré vytvárajú funkcionalitu, ktorú komunita používateľov zvolila za vhodnú na riešenie určitej sady problémov. Okrem úrovní sa jazyk delí aj

na verzie, v ktorých sú definované menšie vývojové zmeny jazyka (upresnenie sémantiky, úpravy v piktogramoch), ale nejde o žiadne veľké zmeny, ktoré by ovplyvňovali, spôsob akým sú SBGN diagramy generované. Nové verzie by mali byť spätne kompatibilné s predchádzajúcimi verziami. To znamená, že nové verzie sú vytvárané v súlade so staršími verziami rovnakej úrovne jazyka, aby boli stále validné, čo však neplatí pre úrovne jazyka.

SBGN je tvorený tromi základných komponentami, ktorými sú *Entity pool nodes* (EPN), *Process nodes* (PN) a *arcs* — hrany. Aby sme si vedeli predstaviť čo PD popisuje a ako vyzerá jeho grafická podoba, použijeme príklad 1.1, ktorý zobrazuje časť MAPK (*kaskáda mitogénom aktivovanéj proteinkinázy*). Väčšie štvoruholníky a kruhy predstavujú biologické objekty, akými sú makromolekuly alebo jednoduché chemikálie, ktoré spoločne nazývame EPN. Tieto objekty sa menia na iné objekty pomocou biochemických procesov, v príklade znázornených ako menšie štvorce — PN. PN a EPN sú navzájom prepojené pomocou šípok (orientovaných hrán). V tomto konkrétnom príklade máme znázornený iba jeden typ PN a tým je proces. Šípky naznačujú smer reakcie, v príklade sa nefosforilovaná RAF kináza, mení na fosforilovanú RAF kinázu v procese katalýzy pomocou RAS katalizátora, čoho vedľajším účinkom je vznik ADP z ATP. V príklade sa vyskytujú aj menšie kruhové objekty, ktoré reprezentujú stavy objektov. Nižšie si detailnejšie popíšeme niektoré typy komponent SBGN-PD.

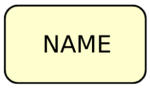
Typy EPN:

- **Nešpecifikovaná entita** — označuje nešpecifikovaný objekt, ktorého typ je neznámy alebo irelevantný pre účely použitia v SBGN diagrame. Znázorňuje sa elipsou (Obr. 1.2).

Obr. 1.2: Príklad nešpecifikovanej entity

- **Makromolekula** — je EPN zahrnutá v mnohých biologických procesoch. Reprezentuje biochemickú látku, zloženú z kovalent-

ných väzieb pseudo-identických jednotiek. Príkladom makromolekúl sú proteíny, nukleové kyseliny, či polysacharidy. Makromolekula je znázornená štvoruholníkom so zaoblenými rohmi (Obr. 1.3).



Obr. 1.3: Príklad makromolekuly

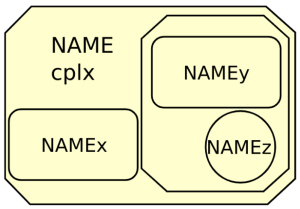
- **Jednoduchá chemikália** — v PD je jednoduchá chemikália opakom makromolekuly. To znamená, že nie je tvorená kovaletnými väzbami. Ide napríklad o atóm, ión, radikál či soľ. Znázorňuje sa jednoduchým kruhom. (Obr. 1.4).

Obr. 1.4: Príklad jednoduchej chemikálie

- **Zdroj a *sink*** — je užitočné mať možnosť znázorniť vznik určitého objektu z nešpecifikovaného zdroja. Ak v modeli vzniká nejaký proteín, nechceme popisovať každú aminokyselinu, cukor a ostatné zložky proteínu. Rovnako užitočné je mať niečo, čím môžme znázorniť deštrukciu alebo rozpad určitej látky, ktorá akoby „zmizla do výlevky“. Pre tieto účely, je v PD definovaný špeciálny piktogram, ktorý reprezentuje neznámy zdroj alebo *sink*. Znázorňuje sa matematickým symbolom prázdnej množiny, kruh prečiarknutý čiarou smerujúcou z pravého horného rohu, do ľavého dolného rohu (Obr. 1.5)

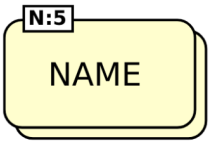
Obr. 1.5: Príklad zdroj/sink

- **Komplex** — zobrazuje biochemický objekt zložený z jednoduchších objektov ako makromolekúl, jednoduchých chemikálií, multimérov alebo aj ďalších komplexov. Výsledkom je nezávislý objekt, ktorý ma svoju vlastnú identitu, vlastnosti a funkciu. Komplex je zobrazený ako štvoruholníkový kontajner, v ktorom sa môžu nachádzať všetky rôzne EPN (vrátane ďalšieho komplexu), s orezanými rohmi (Obr. 1.6).



Obr. 1.6: Príklad komplexu

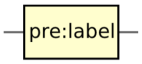
- **Multimér** — je agregácia viacerých identických alebo pseudoidentických objektov združených nekonvalentými väzbami. Multimér sa od polyméru líši práve v tom, že neobsahuje kovalentné väzby, ktoré sú reprezentované makromolekulami. Pod pojmom pseudo-kovalentné je myslené, že entity sa líšia chemicky, ale zdieľajú určitú spoločnú globálnu charakteristiku ako štruktúru či funkciu. Multimér je zobrazený dvoma prekrytými štvoruholníkmi s informáciou o množstve objektov zobrazenom v menšom štvoruholníku (Obr. 1.7).



Obr. 1.7: Príklad multiméru

Doplňujúce štruktúry EPN:

- **Dodatočná informácia** — táto štruktúra predstavuje dodatočné informácie o funkcii danej entity, nesúvisiacej s jeho rolou v SBGN diagrame. Môže napríklad obsahovať informáciu charakterizujúcu logickú časť entity v oblasti funkcionality. Táto informácia sa zobrazuje štvoruholníkom na hranici uzlu entity, ktorú anotuje (Obr. 1.8).



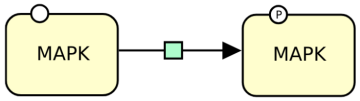
Obr. 1.8: Príklad informácie

- **Stav** — veľa biochemických objektov sa v organizme vyskytuje v rôznych stavoch, ktoré sa môžu meniť za rôznych špecifických podmienok. Stav sa môže meniť napríklad po syntéze, keď zvyšky makromolekúl (aminokyseliny, glycidy, ...) sú kovalentnými väzbami naviazané na iné chemické objekty. Každá entita môže obsahovať jeden alebo viac stavov. Stav sa znázorňuje elipsou nachádzajúcou sa na hranici EPN, ktorého stav popisuje (Obr 1.9).

Obr. 1.9: Príklad stavu

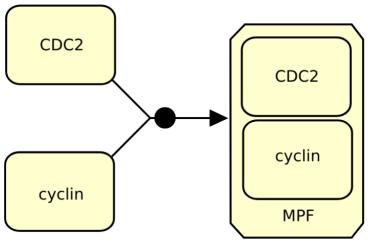
Typy PN:

- **Process** — je základný PN. Popisuje proces transformácie súboru biochemických objektov, na súbor iných biochemických objektov. Transformácia môže znamenať konverziu (modifikáciu kovalentými väzbami), konformáciu (zmenu relatívnej polohy) alebo translokáciu (presun z jedného kompartmentu do iného). Proces je znázornený štvorčekom, na ktorý sú napojené dva konektory (krátke hrany) po jednej na protiľahlých stranách, na ktoré sa napájajú hrany *consumption* a *production*, ktoré budu popísane nižšie. V príklade 1.10 je znázornená fosforylácia proteínu MAP kinázy.



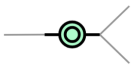
Obr. 1.10: Príklad procesu

- **Association** — tento PD definuje vznik komplexnejšieho objektu, z niekoľko jednoduchších objektov nekonvalentými väzbami. Znázorňuje sa čiernym kruhom s dvoma konektormi, na ktoré sa viažu hrany (consumption a production) (V príklade Obr. 1.11 je znázornené *zjednotenie cyklínu a CDC2 kinázy do MPF komplexu*). Zjednotenie nie je reverzibilné — opačný proces zobrazuje *dissociation*, definovaný nižšie.



Obr. 1.11: Príklad association

- **Dissociation** — popisuje rozpadnutie nekovalentých väzieb a rozklad zložitejších komplexov na jendoduchšie — opačný proces k *association*. Znázorňuje sa podobne ako dissociation, krúžok však nie je vyplnený a navyše obsahuje jeden zanorený kruh (Obr. 1.12).



Obr. 1.12: Príklad dissociation

Hrany:

- **Consumption** — tieto hrany znázorňujú, že objekty zobrazené pomocou EPN vstupujú do určitého biologického procesu. Smerujú z EPN do PN a zobrazujú tak smer reakcie. Znázorňujú

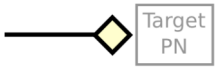
sa jednoduchou rovnou čiarou bez znakov (Obr. 1.13). Môžu obsahovať malý štvorec, definujúci stechiometriu. Každá hrana spája práve jednu EPN s jedným PN.

Obr. 1.13: Príklad consumption hrany

- **Production** — tieto hrany naopak znázorňujú vznik nových objektov — produktov. Rovnako ako pri *consumption* hranách, práve jedna hrana smeruje z PN do EPN. Znázorňujú sa podobne ako *consumption* hrany, ale navyše obsahujú na konci hrany hrot šípky (Obr. 1.14).

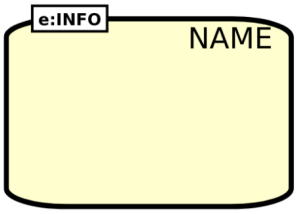
Obr. 1.14: Príklad production hrany

- **Modulation** — hrany znázorňujú proces modulácie cieľového procesu iným objektom či už pozitívne, negatívne alebo oboje, v závislosti na podmienkach. Znázorňuje sa rovnou čiarou s diamantom na konci, smerujúcou k PN, ktorý modifikuje (Obr. 1.15).



Obr. 1.15: Príklad modulation hrany

Kompartment — reprezentuje logickú alebo fyzickú štruktúru, obsahujúcu jeden alebo viacero EPN. Jeden objekt môže vždy patriť iba do jedného kompartmentu. Preto dva „identické“ objekty, patriace do dvoch rôznych kompartmentov, sú v skutočnosti dva rozdielne objekty a mali by byť reprezentované odlišnými objektmi. Kompartment je zobrazený ako uzavretý, spojitý priestor, rôzneho tvaru. (Obr. 1.16).



Obr. 1.16: Príklad kompartmentu

Popísali sme si väčšinu štruktúr, ktoré vytvárajú PD jazyk a ktoré využijeme v ďalšej časti práce. Na to, aby sme sa dostali k jadru problému, si ešte musíme predstaviť jeden formát a tým je BCSL.

1.4.4 BCSL

BCSL [4] bol vytvorený členmi laboratória SYBILA. Je dôležité si uvedomiť rozdiel medzi BCS a BCSL. Hlavnou úlohou BCS, je vytvoriť biologický základ pre matematické modely. BCS je tvorený databázou biochemických objektov, ktoré sú všetky dôkladne anotované a obsahujú odkazy na externe zdroje a databázy, čo umožňuje BCS vytvoriť priestorovú hierarchiu jednotlivých objektov s ich atribútmi a vlastnosťami. Samotné objekty a priestorová hierarchia však na popis modelov nestačia, a je nutné zadefinovať pomocou interaktívnych schém aj hierarchiu procesov, a opísať tak jednotlivé atomické operácie prebiehajúce na entitách.

Na zlepšenie použiteľnosti a presnosti BCS vznikol BCSL, ktorý formálne popisuje objekty a pravidlá. Sémantika jazyka je založená na princípoch modelovania pomocou pravidiel a jedným z dôvodov vzniku bol cieľ vytvoriť jazyk, ktorý by bol ľahko čitateľný nie len pre rôzne nástroje, ale aj pre človeka. BCSL formálne definuje celý BCS a umožňuje definovať všetky objekty a pravidlá, so všetkými ich vlastnosťami. Pomáha prepájať objekty matematických modelov s objektmi BCS, čo vďaka dôkladnej anotácii obohacuje model o ďalšie informácie. Model definovaný v BCSL má dve výhody — možnosť aplikovať formálnu analýzu na model a úplne zrekonštruovať biologický význam celého modelu.

Formálny základ jazyka BCSL tvoria pravidlá, v ktorých vystupujú agenti. Definujeme konkrétne tri typy agentov — atomický, štruktúrny

a komplexný. Taktiež definujeme signatúru a kompartment.

Signatúra môže byť atomická alebo štruktúrna. Atomická obsahuje množinu možných stavov, v ktorých sa môže atomický agent nachádzať a štruktúrna obsahuje názvy atomických agentov, ktoré sa môžu v štruktúrnom agentovi vyskytovať.

Atomický agent je najjednoduchšia jednotka BCSL, slúžiaca na popis biochemických objektov. Atomický agent sa vždy nachádza v nejakom stave. V prípade, že stav nie je dôležitý, alebo je neznámy definujeme „prázdny stav“, ktorý označujeme symbolom ϵ (epsilon).

Povolené stavy agenta definuje signatúra s rovnakým menom, akým je meno atomického agenta. Atomický agent sa definuje ako dvojica obsahujúca meno agenta a jeho stav. Príklad zápisu atomického agenta:

chl{n}

kde *chl* je názov agenta a *n* je jeho stav.

Štruktúrny agent popisuje biochemické objekty, ktoré sú zložene z viacerých menších objektov (atomických agentov), pričom zloženie je iba abstraktné. Štruktúrny agent má unikátne meno a môže byť zložený len z atomických agentov, ktoré sa nachádzajú v rovnakom kompartmente ako štruktúrny agent samotný.

Typickým príkladom štruktúrnych agentov sú proteíny, pozostávajúce z aminokyselín. Aminokyselín sa v proteíne môže nachádzať stovky, ale nie všetky nás v modely zaujímajú. Povedzme, že len dve aminokyseliny sú schopné určitej modifikácie (zmeny stavu), ktorá nás zaujíma. Je teda logickejšie zachytiť v štruktúrnom agentovi len tieto dve aminokyseliny a zvyšnú štruktúru proteínu abstrahujeme.

Štruktúrny agent je definovaný ako dvojica pozostávajúca z mena štruktúrneho agenta a množiny atomických agentov, ktoré sa v ňom nachádzajú. Príklad štruktúrneho agenta obsahujúceho troch atomických agentov:

ps2(chl{n}|p680{+}|pheo{-})

kde *ps2* je názov štruktúrneho agenta a *chl, p680, phe* sú atomický agenti.

Komplexný agent — definuje zložitejšie netriviálne objekty, zložene zo všetkých tipov agentov, vrátane komplexných. V SBML a SBGN boli komplexy charakterizované pomocou väzieb. BCSL sa od toho snaží abstrahovať a pod komplexom myslí existenciu určitých objektoch v akomsi zhluku či skupine, ktorý zdieľa určité vlastnosti. Komplex definujeme ako dvojicu obsahujúcu množinu agentov, ktorí komplex vytvárajú a kompartment v ktorom sa nachádza. Jednotlivé zložky komplexu sú v zápise oddelené bodkou. Príklad komplexu:

$$ps2(chl\{n\}|p680\{+\}|pheo\{-\}) \,.\, fa\{-\}::tlm$$

V príklade je komplex tvorený jedným štruktúrnym agentom *ps2*, jedným atomickým agentom *pheo* a samotný komplex sa nachádza v kompartmente *tlm*

Každý samostatne stojací objekt, ktorý je v pravidle oddelený znamienkom plus, (jednotlivé reaktanty a produkty v reakcii), musí byť lokalizovaný v nejakom kompartmente. Všetci zvyšní agenti, z ktorých je hlavný objekt zložený, sa musia nachádzať v rovnakom kompartmente. Kompartment sa v zápise objektu vždy nachádza na poslednom mieste a je oddelený od zvyšku agenta dvomi, po sebe nasledujúcimi dvojbodkami (NADPH::cyt).

Pravidlo je tvorené z jednotlivých agentov a kompartmentov. Každé pravidlo má dve strany, ktoré však nie vždy obsahujú nejakých agentov (vznik a zánik určitých objektov). Jednotlivé strany pravidla sú oddelené šípkou, buď $\Rightarrow$ — čo indikuje, že pravidlo je ireverzibilné alebo $\Leftrightarrow$ pre reverzibilné pravidlá. Samotná formálna definícia je zložitejšia, avšak nie až tak dôležitá pre koncept tejto práce a názorný príklad pravidla bude postačujúci pre jeho pochopenie.

$$HCO_3\{-\} :: liq + OH\{-\} :: liq \Leftrightarrow CO_3\{2-\} :: liq + H_2O :: liq$$

V pravidle sa nachádzajú dva reaktanty — ($HCO_3\{-\}$, *OH{-}*) a dva produkty — ($CO_3\{2-\}$, H_2O). Všetci agenti sa nachádzajú v rovnakom kompartmente — *liq* a $\Leftrightarrow$ naznačuje, že pravidlo je reverzibilné.

H_2O je v prázdnom stave ϵ, čo sa v zápise môže vynechať.

Model je v BCSL definovaný ako štvorica, obsahujúca pravidlá, atomickú signatúru, štruktúrnu signatúru a počiatočné podmienky modelu, ktoré definujú stav modelu na začiatku. Správanie sa modelu je definované množinou pravidiel. Atomická signatúra definuje pre každého atomického agenta, ktorý sa v pravidlách nachádza, všetky stavy ktoré môže nadobúdať. Podobne štruktúrna signatúra definuje pre každého štruktúrneho agenta, ktorý sa v pravidlách vyskytuje, atomických agentov, z ktorého môže byť tvorený.

Syntaktické rozšírenia

- V prípade, že štruktúrny agent obsahuje atomických agentov len s prázdnym stavom ϵ, je možné tieto stavy z pravidla vynechať.
- Okrem atomických a štruktúrnych signatúr je možné si zadefinovať aj komplexnú signatúru, ktorá obsahuje unikátne mená (alias), komplexných agentov, ktorými môžme v pravidle konkrétny komplexný agent substituovať a nezapisovať tak celú jeho vnútornú štruktúru.
- Stechiometria — definuje množstvo jednotlivých reaktantov a produktov v pravidle, nachádza sa pred agentom.
- Operátorom „::“, ktorý je použitý pre priradenie agenta ku kompartmentu, je tiež možné naznačiť priradenie agenta do komplexu, čo umožňuje prehľadnejšie zanorovanie sa, hlbšie do biochemických štruktúr (*S{u}::KaiC::KaiC3::cyt*).
- Posledným rozšírením, je možnosť si zadefinovať v pravidle premennú, ktorá v pravidle reprezentuje ľubovoľný objekt z danej množiny, čo umožňuje zredukovať počet podobných pravidiel.

  S{u}::KaiC::?X::cyt ⇒ S{p}::KaiC::?X::cyt ; ?X = {KaiC3, KaiBC}

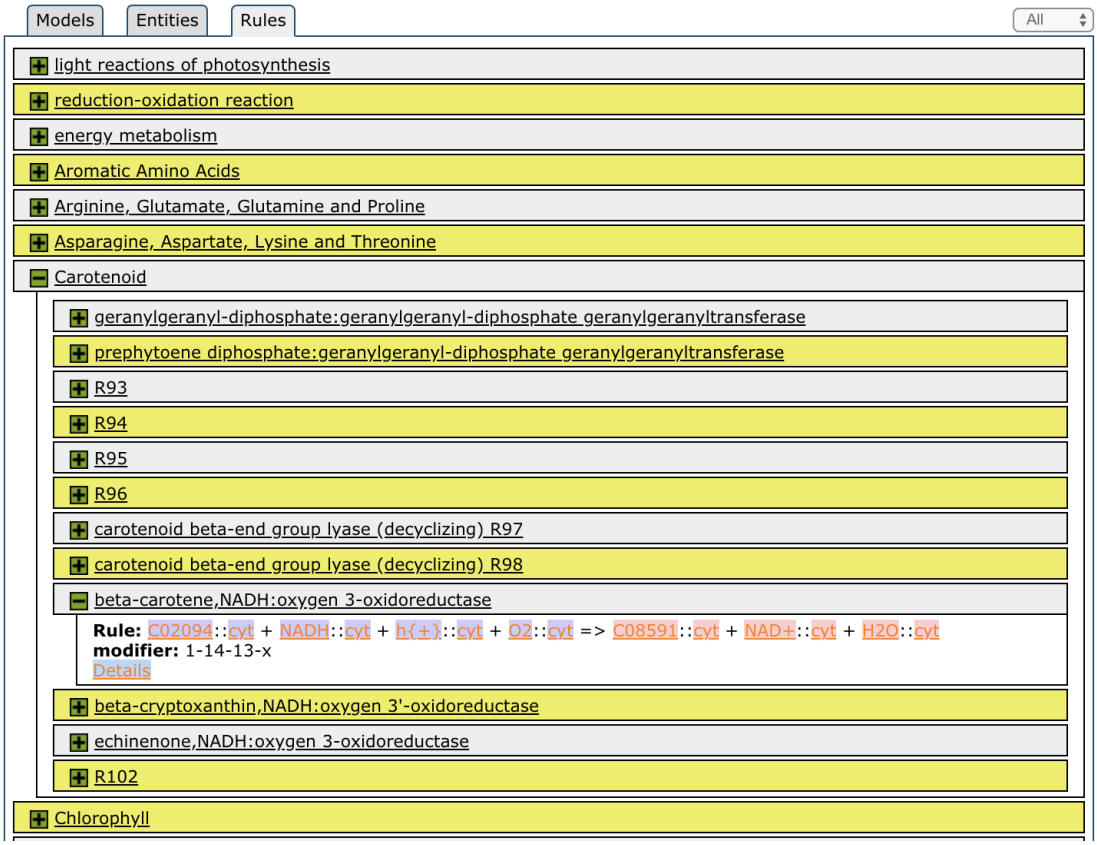
  V tomto príklade je zadefinovaná premenná X, ktorej množina možných hodnôt sa nachádza za pravidlom. X môžme v tomto prípade substitovať za KaiC3 alebo KaiBC. Ak by premenná

nebola zadefinovaná, muselo by byť toto pravidlo rozpísané na dve samostatné pravidla. (V skutočnosti to však reprezentuje dve odlišné pravidla, tento zápis nám len pomáha zredukovať počet pravidiel a tým zjednodušiť zápis modelu).

2 Popis problému

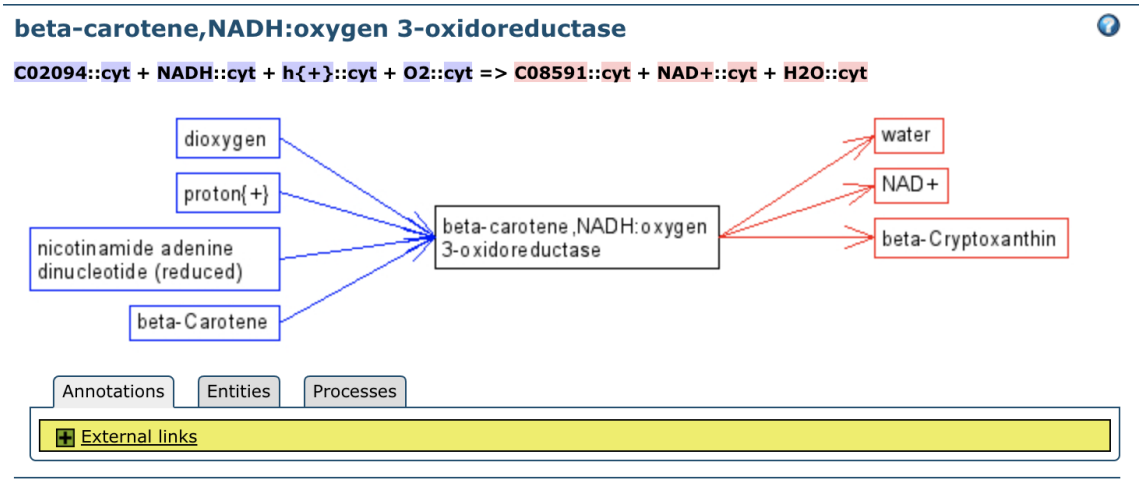
V predchádzajúcej kapitole sme si popísali všetky potrebné pojmy a definície. V tejto kapitole si popíšeme problém, ktorý je objektom tejto práce. Vizualizácia modelov na portáli e-cyano je v čase vzniku tejto práce v zastaralom a improvizovanom stave. Cieľom tejto práce bolo vymyslieť a neimplementovať lepší spôsob vizualizácie modelov, so zameraním na pravidla v module *BCS* a reakcie v *Model repository*, ktorý bude použitý pre účely CMP.

V čase vzniku práce je stav nasledovný. Ak si na portáli v BCS module, zvolíme sekciu pravidiel niektorého z modelov, objaví sa nám zoznam všetkých pravidiel (Obr. 2.1).



Obr. 2.1: Zoznam BCS pravidiel na portáli e-cyano

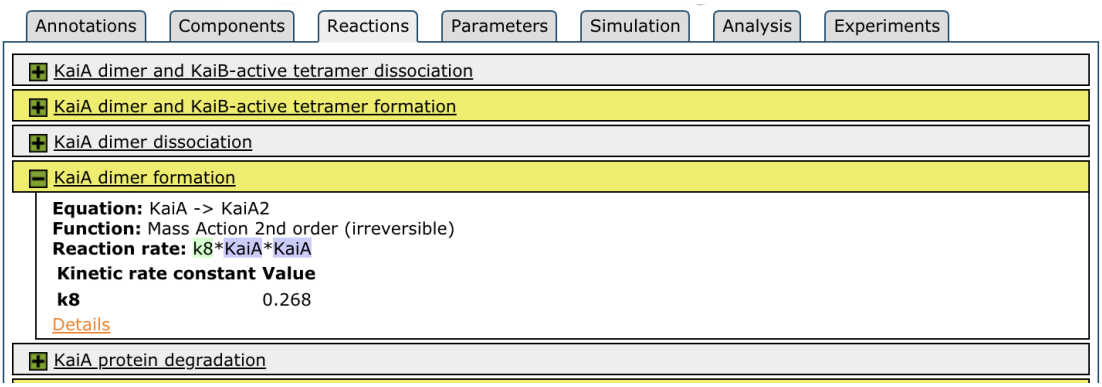
V prípade, že si vyberieme niektoré konkrétne pravidlo a rozbalíme záložku, zobrazí sa nám pravidlo, modifikátor a tiež možnosť kliknúť na detaily. Ak zvolíme detaily, presmeruje nás to na detaily pravidla, kde je samotné pravidlo aj vizualizované (Obr 2.2).



Obr. 2.2: Detaily konkrétneho BCS pravidla na portáli e-cyano

Ako môžme vidieť, vizualizácia pravidla je sprostredkovaná len pomocou jednoduchých farebných štvoruholníkov a šipiek. Schéma neznázorňuje štruktúru pravidla, nie je možné identifikovať či ide o atomického, štruktúrneho či komplexného agenta a taktiež chýba informácia o kompartmente. Úlohou práce je tento spôsob vizualizácie nahradiť a automaticky vizualizovať pomocou vhodných nástrojov v štandarde SBGN.

Pre reakcie reprezentované v štandardu SBML, nachádzajúce sa v *Model repository* (Obr. 2.3), sa možnosť vizualizácie nenachádza vôbec. V prípade, že si zvolíme detaily reakcie, zobrazia sa detaily BCS pravidla, na ktoré je reakcia mapovaná a to je opísané vyššie.



Obr. 2.3: Zoznam reakcií v Repozitore modelov na portáli e-cyano

V obidvoch prípadoch, ako pre reakcie tak aj pravidlá, bolo potrebné vytvoriť spôsob automatickej vizualizácie v štandarde SBGN. Pomocou SBML je možné definovať omnoho komplexnejšie modely, aké je pomocou SBGN možné zobraziť.

V tomto prípade ide iba o vizualizovanie jednoduchších SBML reakcií, čo nepredstavuje taký komplexný problém, akým by bolo vizualizovanie celého SBML modelu. Jednotlivé látky (reaktanty, produkty a modifikátory) SBML reakcií, sú v SBGN reprezentované pomocou EPN makromolekúl a proces zmeny reaktantov na produkty je zachytený pomocou PN *proces*.

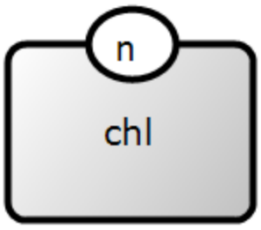
S BCSL pravidlami je to zložitejšie. Mapovanie BCS na SBGN zatiaľ nie je známe — doposiaľ nebolo nikde definované. Preto jednou z kľúčových úloh tejto práce je vytvoriť návrh mapovania z BCSL do SBGN.

Ďalšou úlohou práce je navrhnúť a implementovať nástroj, ktorý bude schopný plne automaticky, bez akéhokoľvek manuálneho ľudského zásahu, spracovať SBML reakciu alebo BCSL pravidlo, vytvoriť validný SBGN súbor a vygenerovať výsledný SBGN diagram v požadovanom formáte (PNG alebo SVG), ktorý vráti na výstup.

3 Návrh mapovania BCSL na SBGN

V tejto kapitole je definovaný spôsob mapovania BCSL pravidiel do SBGN štandardu. Ako bolo spomenuté v predchádzajúcej kapitole, prevod z BCSL do SBGN doposiaľ nebol nikde definovaný a nasledujúci návrh mapovania, je jeden z možných spôsobov, akým by bolo možné tuto konverziu uskutočniť.

Začneme od základného BCSL objektu, ktorým je atomický agent. Skutočnosť, že atomický agent sa nachádza vždy v určitom stave (prípadne prázdny stav — ϵ) a že vždy vieme o aký objekt ide, keďže BCS kladie dôraz na dôkladnú anotáciu jednotlivých objektov, môžme vylúčiť použitie jednoduchej chemikálie (ktorá v SBGN diagrame nemôže obsahovať stav), a taktiež nešpecifikovanú entitu. Pre atomických agentov sa ako najvhodnejší EPN objekt z SBGN javí makromolekula s dodatočnou informáciou o stave. Povedzme, že mame atomického agenta *chl{n}*. Jeho grafická podoba v SBGN by vyzerala nasledovne:

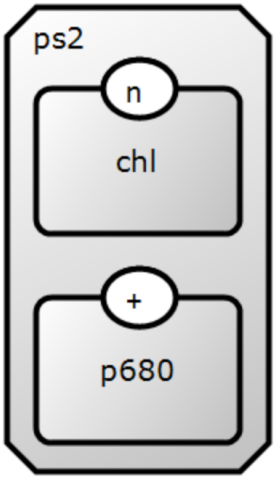


Obr. 3.1: Príklad mapovania atomického agenta do SBGN

V prípade, že obsahuje prázdny stav, piktogram stavu sa v zobrazení EPN vynechá.

Ďalším objektom BCSL je štruktúrny agent. Vzhľadom k tomu, že štruktúrny agent je tvorený z jednoduchších atomických agentov, čo naznačuje určité zanorenie, ako jediná možnosť sa javí reprezentovať štruktúrneho agenta v SBGN pomocou EPN komplexu. Komplex bude v ľavom hornom rohu obsahovať informáciu o názve štruktúrneho agenta a vo vnútri SBGN komplexu budu rozmiestnení jednotliví atomickí agenti, reprezentovaní EPN makromolekulami. Uvažujme

štruktúrneho agenta *ps2(chl{n}|p680{+})*. Zobrazený v SBGN by vyzeral nasledovne:



Obr. 3.2: Príklad mapovania štruktúrneho agenta do SBGN

Posledným agentom je komplexný agent. Keďže podobne ako štruktúrny agent obsahuje zanorené objekty, (a to aj hlbšie ako do jednej úrovne zanorenia), je tiež najvhodnejšie použiť pre jeho grafické zobrazenie v SBGN komplex. Narozdiel od štruktúrneho agenta, komplex neobsahuje názov, čo sa hodí k rozlíšeniu, či ide o štruktúrneho, alebo komplexného agenta.

Ďalším rozpoznávacím prvkom môže byť to, že štruktúrny agent môže obsahovať len atomických agentov (čiže EPN makromolekuly), zatiaľ čo komplexný agent môže obsahovať všetky ostatné typy agentov. To však nie je až tak vhodný rozpoznávací prvok, na ktorý sa nemôžeme spoliehať v prípade, že komplex je tiež tvorený len atomickými agentmi. Stále je možné rozlíšiť, že ide o komplex absenciou názvu.

V prípade, že máme definovanú komplexnú signatúru, čo znamená že určitý komplexný agent môže byť v BCSL pomenovaný, bude pomenovaný aj SBGN komplex, ale nebude obsahovať žiadne zanorené objekty vo vnútri (informácie o vnútornej štruktúre komplexu sa v komplexnej signatúre nachádzajú).

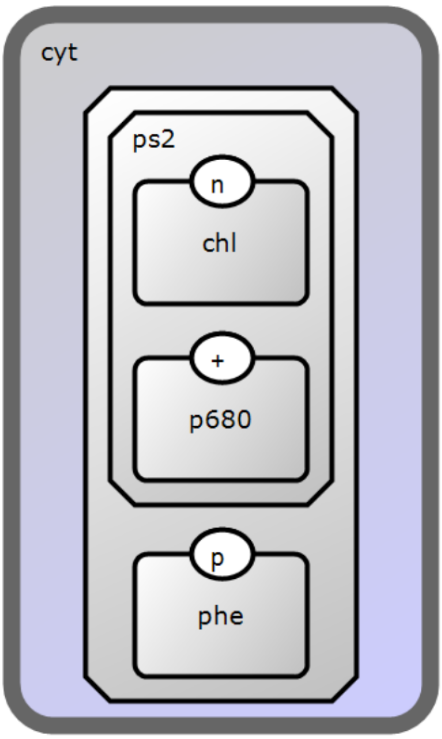
Rozlíšiť komplexný a štruktúrny agent graficky pomocou rozdielnych SBGN objektov v súčastnosti nie je možné, keďže komplex je

jediný objekt z SBGN, ktorý môže obsahovať iné objekty. Pre jazyky založené na pravidlách je grafická notácia v SBGN nejasná a najlepším riešením by bolo ju rozšíriť, čo by vyžadovalo zásah samotných autorov a editorov jazyka.

Ak by však vyššie spomenuté riešenie s absenciou mena komplexu ako rozlišovacie znamenie bolo nedostatočné, jednou z možností ako striktnejšie definovať, že ide o komplex, je pridať „Komplex“ k názvu. Pre komplexy, ktoré nemajú pomenovanie použiť „Komplex“ ako názov a pre pomenované komplexy pridať „Komplex – “ ako prefix.

Pre zobrazenie samotného kompartmentu je v SBGN konkrétny objekt, a tak kompartment by nemal predstavovať žiadne komplikácie.

Uvažujme BCSL komplex *ps2(chl{n}|p680{+}).phe{p}::cyt*. Grafické zobrazenie aj s kompartmentom v SBGN by mohlo vyzerať nasledovne:



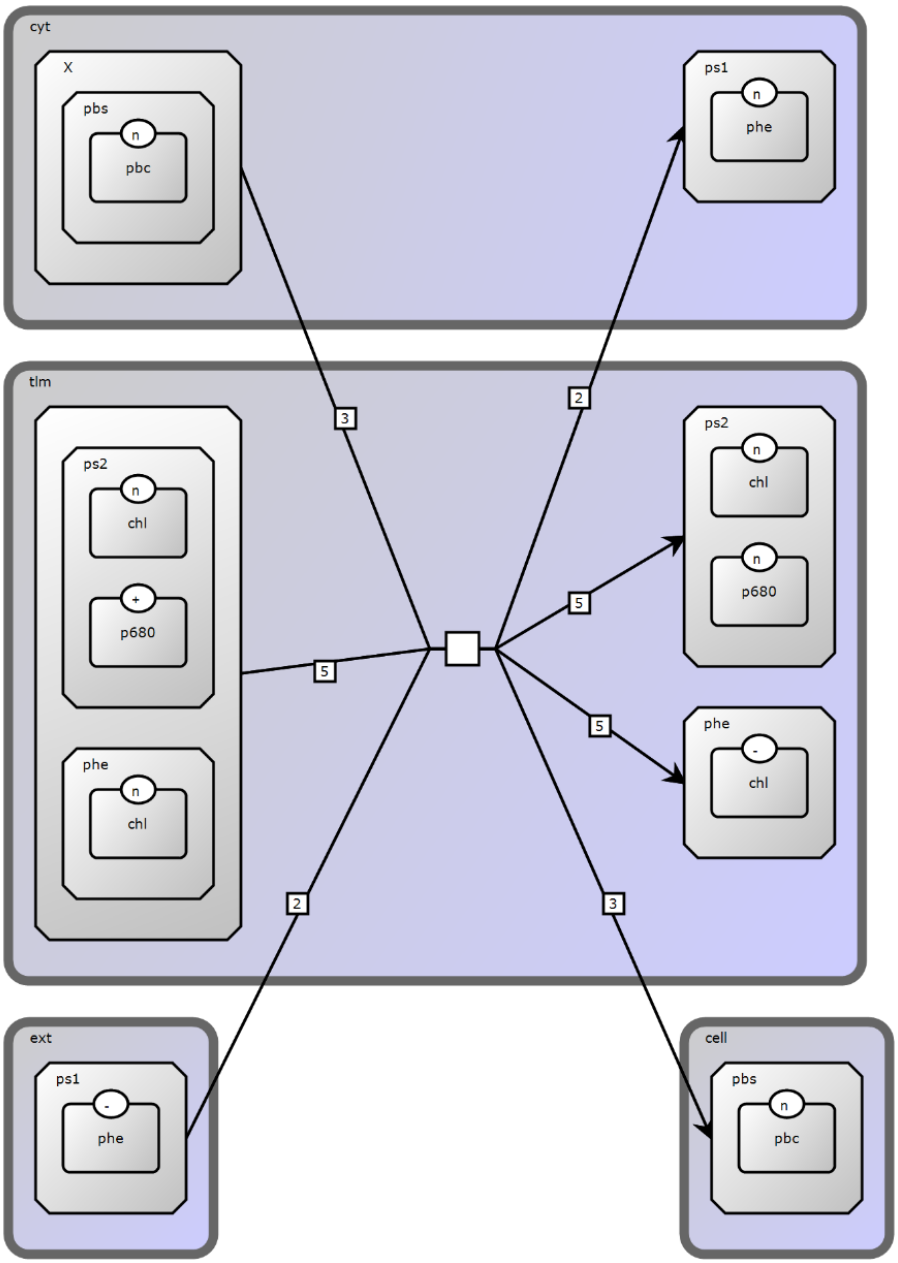
Obr. 3.3: Príklad mapovania komplexného agenta do SBGN

BCSL pravidlá definujú správanie modelu a teda zmenu vstupných objektov (reaktantov) na výstupné objekty (produkty). Na zobrazenie tejto zmeny je v SGBN je najvhodnejšie využiť PN proces. Stechiometriu je možné zobraziť pomocou štvoruholníkových dekorátorov, ktoré sa pridávajú na hrany smerujúce z EPN, zastupujúce jednotlivé objekty, do PN. V nasledujúcom príklade (Obr. 3.4) je zobrazené vymyslené pravidlo (slúžiace len na vizuálnu ukážku komplexnosti vytvoreného mapovania).

V príklade 3.4 je zobrazené pomerne zložité pravidlo,

$$
\begin{gathered}
3\ \ pbc\{n\} :: pbs :: X :: cyt\ \ + \\
2\ \ phe\{-\} :: ps1 :: ext\ \ + \\
5\ \ ps2(chl\{n\}|p680\{+\}).phe(chl\{n\}) :: tlm \\
\Rightarrow \\
3\ \ pbc\{n\} :: pbs :: cell+ \\
2\ \ phe\{n\} :: ps1 :: cyt\ \ + \\
5\ \ ps2(chln|p680n) :: tlm\ \ + \\
5\ \ phe(chl-) :: tlm
\end{gathered}
$$

v ktorom sa nachádzajú 4 rozdielne kompartmenty, prechod EPN z jedného kompartmentu do druhého, rozpad EPN, viacnásobné zanorenie komplexov a taktiež je zahrnutá aj stechiometria.



Obr. 3.4: Príklad zobrazenia kompletného BCSL pravidla do SBGN

4 Rešerš možných nástrojov a aplikácií

V tejto kapitole si predstavíme zopár nástrojov, knižníc a aplikácií slúžiacich na vytváranie, manipuláciu, vizualizáciu SBGN dát.

4.1 Cytoscape

Cytoscape je open source desktopová aplikácia na vizualizáciu molekulárnych interakcií a biologických sietí. Cytoscape bol vyvinutý prevažne pre biologické účely a výskum, ale je to platforma pre všeobecnú komplexnú sieťovú analýzu a vizualizáciu. Ponúka veľa funkcií pre integráciu, analýzu a vizualizáciu dát.

4.2 MINERVA

MINERVA [9] je webová platforma slúžiaca na vizualizáciu a manažment molekulárnych interakčných sieti uložených v SBGN formáte. MINERVA poskytuje automatickú anotáciu a verifikáciu dát. Funkcie exportu umožňujú sťahovanie konkrétnych oblastí vizualizovaných sietí ako modely kompatibilné so štandardom SBGN, pre efektívne opätovné použitie. Vizualizácia je generovaná s pomocou Google maps *Application Programming Interface* (API).

4.3 CySBGN

CySBGN [10] je dodatočný balíček, ktorý si užívateľ môže stiahnuť a nainštalovať do nástroja Cytoscape. CySBGN umožňuje importovanie, modifikovanie a analýzu SBGN máp v Cytoscape. Ponúka širokú paletu nástrojov dostupných v tejto platforme pre prácu so sieťami vo formáte SBGN. CySBGN podporuje všetky tri jazyky štandardu SBGN a umožňuje automatickú generáciu SBGN diagramov z importovaného SBML modelu. Bohužiaľ, CySBGN nepodporuje všetky typy hrán a uzlov z SBGN-PD.

4.4 Biographer

Biographer [11] je webový editor pre reakčné siete, ktorý môže byť integrovaný ako knižnica do nástrojov, ktoré pracujú s informáciami zaoberajúcimi sa sieťami. Softvér umožňuje vizualizáciu založenú na štandarde SBGN. Je schopný importovať siete v rôznych formátoch ako sú SBML, SBGN a jSBGN (vlastný formát nástroja Biographer na zdieľanie dát). Obsahuje interaktívne nástroje na úpravu grafov a algoritmy automatického rozloženia diagramu. Poskytuje samostatný editor grafov a webový server, ktorý obsahuje rozšírené funkcie ako sú webové služby pre import a export modelov a vizualizácie v rôznych formátoch. Domovská stránka projektu je však nanešťastie nedostupná a nie je možné sa dostať k obsahu. Zdá sa, že vývoj a podpora nástroja skončila.

4.5 VISIBIOweb

VISIBIOweb [12] je bezplatná webová služba, ktorá sa používa na vizualizáciu a rozloženie modelov v BioPAX formáte (jazyk podobný SBML, ktorého cieľom je umožniť integráciu, výmenu, vizualizáciu a analýzu údajov o biologických sietiach). Je vytvorená v jazyku JavaScript a podobne ako MINERVA využíva Google maps API a zobrazuje statický obrázok v štandarde SBGN-PD. Importovať je možné iba súbory typu „.owl“, avšak používatelia sú schopní exportovať grafy v SBGN. Poskytuje službu automatického rozloženia diagramu. Táto služba prijíma biologické siete v jednoduchom formáte XML a vráti ich späť klientovi v rovnakom formáte s pridaním informácií o rozložení (vygeneruje súradnice pre všetky SBGN komponenty), čo by sa pre koncept našej práce presne hodilo. Avšak sú tu dva problémy. Prvým je, že automatické generovanie nie je moc presné a komponenty sa často prekrývajú. Druhým, závažnejším problémom, je skutočnosť, že podobne ako Biographer, domovská adresa služby je nedostupná a vyzerá to tak, že podpora tejto služby taktiež skončila.

4.6 SBGNViz

SBGNViz [13] je webový prehliadač pre diagrami procesov v SBGN-PD. SBGNViz dokáže zobraziť formáty BioPAX aj SBGN. Jedinečné vlastnosti SBGNViz zahŕňajú schopnosť zanoriť uzly do ľubovoľnej hĺbky, aby dôkladne reprezentovali molekulárne komplexy, automatické rozloženie jednotlivých komponent, editovanie a možnosť prezrieť si podrobné informácie o entitách. SBGNViz môže byť použitý vo webovom prehliadači bez akejkoľvek inštalácie a môže byť ľahko implementovaný do webových stránok. SBGNViz má dve verzie postavené pomocou jazyka ActionScript a JavaScript. Importovať je možné len validný SBGN súbor vo formáte XML, exportovanie je možné aj vo formátoch XML, PNG alebo JPG.

4.7 LibSBGN

LibSBGN [14] je oficiálna softvérová knižnica pre čítanie, zapisovanie a manipuláciu s SBGN dátami. Knižnica (dostupná v jazykoch C++, Java a Python) uľahčuje vývojárom pridávanie SBGN nástrojov, zatiaľ čo formát súborov uľahčuje výmenu SBGN dát medzi kompatibilnými softvérovými aplikáciami. Knižnica tiež podporuje validáciu SBGN dát, čo zabezpečuje, že výsledné diagramy sú v súlade s podrobnými špecifikáciami SBGN. Knižnica umožňuje nie len vytváranie a zapisovanie SBGN máp vo formáte XML, ale taktiež vygenerovanie samotných SBGN diagramov vo formáte PNG, za pomoci externého API.

Pomocou jednotlivých tried a metód je možné vytvoriť SBGN objekt, ktorý obsahuje všetky informácie o podobe výsledného diagramu, ako je jeho celková šírka a dĺžka diagramu či informácie o jednotlivých komponentách, ktoré majú prísne dané súradnice, dĺžku a šírku. Knižnica umožňuje exportovať SBGN vo formáte XML, ktorý je vhodný na uloženie a zdieľanie obsahu s možnosťou výsledný SBGN súbor ďalej modifikovať (vygenerovaný obrázok už nie je možné meniť).

4.8 Krayon

Krayon [15] je desktopová aplikácia vyvíjaná v jazyku Kotlin, s príjemným užívateľským rozhraním. Slúži ako interaktívny editor pre SBGN diagramy. Umožňuje export SBGN diagramu v rôznych formátoch (PDF, PNG, SVG, JPG, GIF), importovať vstupné dáta je však možné len vo formáte SBGN. Poskytuje jednoduchý interaktívny režim, ktorý pomôže rýchlo vytvoriť platný SBGN diagram.

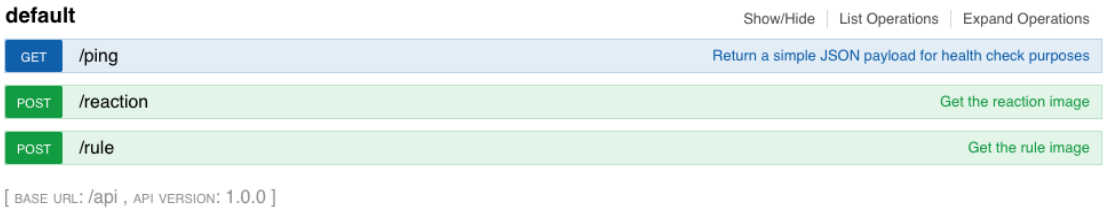
5 Návrh a implementácia aplikácie

V predchádzajúcej kapitole sme si uviedli niekoľko nástrojov, slúžiacich na prácu a vizualizáciu SBGN dát. Hlavný nedostatok väčšiny spomenutých nástrojov je, že buď ide o online webové platformy alebo desktopové aplikácie, ktoré potrebujú manuálny zásah na vytváranie alebo upravovanie SBGN diagramov. Väčšina z nich taktiež na vstup potrebuje už validný SBGN súbor vo formáte XML, ktorý my však nemáme. Vhodnými nástrojmi by boli Biographer alebo VISIBIOweb, ktoré poskytovali API na automatické vytváranie rozloženia jednotlivých komponent v SBGN diagramoch, čo z implementačného hľadiska predstavuje najkomplexnejší problém tejto práce, keďže každá SBGN komponenta obsahuje svoje vlastné súradnice, výšku a šírku, čo sa prejaví na výslednej podobe diagramu.

Správne rozloženie týchto komponent zohráva kľúčovú úlohu v pochopení jednotlivých pravidiel a reakcií, čo koniec koncov predstavovalo hlavný cieľ tejto práce — aby užívateľské rozhranie CMP bolo obohatené o kvalitnejšiu a prehľadnejšiu vizualizáciu biochemických dát a tým viedlo k rýchlejšiemu a jednoduchšiemu pochopeniu modelových dát. Vzhľadom k tomu, že nástroje Biographer a VISIBIOweb sú „mŕtve“ a zdá sa, že ich podpora skončila, nie je možné využiť ich služby prostredníctvom API. Najvhodnejším riešením sa pre účely práce javilo použitie oficiálnej LibSBGN knižnice, ktorá má podporu v troch populárnych programovacích jazykoch. S využitím e-cyano API, ktoré poskytuje informácie o jednotlivých objektoch vyskytujúcich sa v reakciách a pravidlách, ďalej s použitím LibSBGN knižnice, syntaktického parseru SBGN pravidiel napísaného v C++ a ďalších nástrojov, bola vytvorená aplikácia, ktorá vytvorí a následne na výstup vráti validný SBGN diagram.

5.1 Návrh aplikácie

Aplikácia mala jeden hlavný cieľ a tým bolo plne automatizované vytvorenie validných SBGN diagramov pre SBML reakcie a BCSL pravidlá. API je preto tvorené z dvoch HTTP POST požiadavok a jedného GET požiadavku (Obr 5.1).



Obr. 5.1: Návrh HTTP požiadavok navrhovaného API, pomocou nástroja Swagger

GET požiadavok s cestou „/ping“ slúži ako indikátor, že API je spustené a riadne beží. Ak zašleme požiadavok na túto cestu ako odpoveď dostaneme jednoduchý JSON objekt, ktorý obsahuje jediný kľúč („pong“) s hodnotou („true“), čo značí, že API je spustené a odpovedá na HTTP požiadavky.

POST metóda s cestou „/reaction“ slúži ako požiadavok na vytvorenie SBGN diagramu pre SBML reackiu. Telo validného POST požiadavku by mal byť JSON, ktorý obsahuje tri kľúče:

- model_id — číslo reprezentujúce ID modelu, v ktorom sa reakcia nachádza
- reaction_id — číslo reprezentujúce ID reakcie z modelu, ktorá má byť vizualizovaná
- as_svg — indikátor obsahujúci hodnotu („true“ alebo „false“), udávajúci či výsledný obrázok má byť vo formáte SVG alebo PNG

Pomocou jednotlivých ID hodnôt a e-cyano API, je aplikácia schopná zistiť o aký model a konkrétnu reakciu modelu ide a získať jednotlivé komponenty reakcie a ich typy, s ktorými ďalej pracuje.

Druhý POST požiadavok s cestou „/rule“ , je požiadavok na vytvorenie SBGN diagramu z BCSL pravidla. Podobne ako predchádzajúce volanie, obsahuje telo volania JSON, ktorý však obsahuje len dva kľúče:

- rule — textový reťazec reprezentujúci samotné pravidlo v BCSL, napríklad: $ps2(qa-|qbn) :: tlm \Leftrightarrow ps2(qan|qb-) :: tlm$

- as_svg — rovnaký indikátor ako v predchádzajúcom volaní

Textový reťazec je pomocou aplikácie a externej knižnice spracovaný do syntaktického stromu, s ktorým potom aplikácia ďalej pracuje.

5.2 Použité nástroje

V tejto podkapitole si popíšeme nástroje, ktoré sú v aplikácii použité. Jedným z použitých nástrojom je spomínaná knižnica LibSBGN.

5.2.1 BCSLruleParser

BCSLruleParser [16] je knižnica napísaná v jazyku C++ a bola vytvorená pre účely Laboratória systémovej biológie (SYBILA) a BCSL. Knižnica slúži na syntaktické spracovanie BCSL pravidla do špecifickej stromovej štruktúry, ktorý zachytáva a popisuje štruktúru pravidla a umožňuje jednoduchšie strojové spracovanie a prácu s BCSL pravidlom. Knižnicu ju možné pomocou mapovania použiť v jazyku PHP a Python.

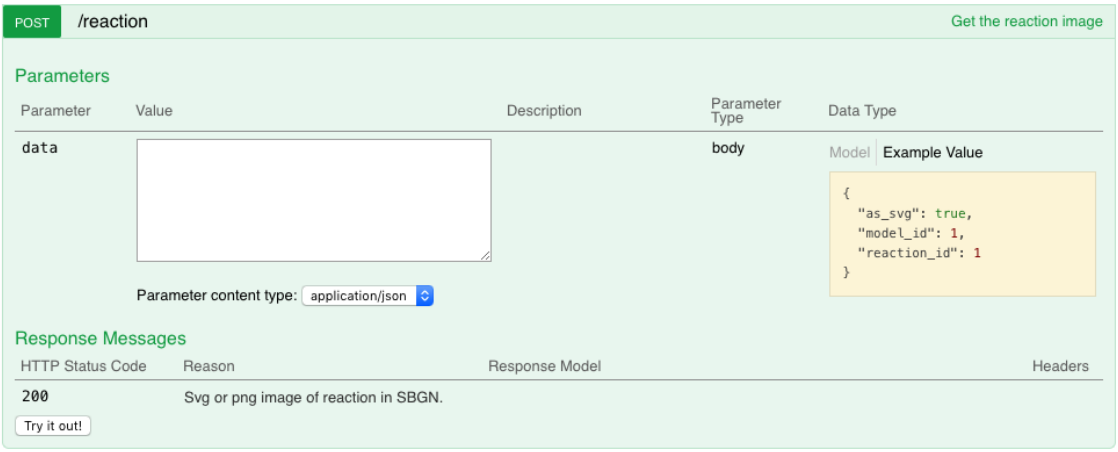
5.2.2 Potrace

Potrace [17] je nástroj na transformovanie bit-mapových formátov (PBM, PGM, PPM alebo BMP) do jedného z niektorých vektorových formátov (SVG, PDF). Typickým príkladom použitia je vytváranie súborov SVG alebo PDF zo skenovaných údajov ako sú firemné alebo univerzitné logá, ručne písané poznámky atď. Výsledný obrázok je následne možné vykresliť v akomkoľvek rozlíšení.

5.2.3 Swagger

Swagger [18] je softvérový nástroj, tvorený veľkým počtom funkcií, ktoré pomáhajú vývojárom navrhovať, vytvárať, dokumentovať a konzumovať RESTful webové služby. Swagger umožňuje jednoduchý návrh aplikácie súčasne s kontrolou vstupných a výstupných dát, automatizovanú dokumentáciou a pomocou rozhrania, ktoré generuje, umožňuje aj možnosť jednotlivé API služby volať a testovať. V príklade (Obr. 5.2) je zobrazené Swagger rozhranie, definujúce HTTP požiadavok na vygenerovanie SGBN diagramu pre konkrétnu reakciu. V ľavo

je popis a typ parametrov API požiadavku a v pravo je príklad dát, s ktorými môžeme API zavolať. Rozhranie tiež zobrazuje typ a cestu API požiadavku, dokumentáciu a popis výstupných dát, to všetko na jednom mieste.



Obr. 5.2: Swagger rozhranie definujúce HTTP požiadavok na generovanie SBGN pre reakciu

5.3 Implementácia

V tejto podkapitole si bližšie popíšeme samotný postup implementácie.

Aplikácia je vytvorená v jazyku Python. Tento jazyk bol zvolený z viacerých dôvodov. Prvým dôvodom bola skutočnosť, že použitá knižnica LibSBGN obsahuje podporu pre Python a je možné s ňou jednoducho pracovať. Druhým dôvod bola knižnica BCSLruleParser, ktorá poskytuje mapovanie do jazyka Python a je možné ju využiť na spracovanie BCSL pravidiel z textového reťazca do jednoduchšej stromovej formy, čo značne zjednodušuje prácu s pravidlami. Tretí dôvod je fakt, že Python je populárny programovací jazyk s veľkým množstvom knižníc, ktorý sa skvelé hodí na vývoj API tohoto druhu. Umožňuje rýchly ako samotný vývoj, tak aj následné zmeny a vylepšenia aplikácií, má jednoduchú prehľadnú syntax, ktorú programátor, ktorý ovláda iní programovací jazyk rýchlo pochopí a s kvalitnou

dokumentáciu a anotáciou kódu predstavuje vhodný jazyk pre vývoj tejto aplikácie. Aplikácia využíva Python 3.6.5. a beží vo virtuálnom prostredí, čo zabezpečuje jednoduchú inštaláciu systémových požiadavok na väčšine populárnych operačných systémoch.

5.3.1 Implementácia pravidiel

Aby sa zachovala správna štruktúra BCSL pravidiel a vytvoril sa validný SBGN diagram, aplikácia využíva viackrát spomínanú stromovú štruktúru pravidla (vytvorenú pomocou knižnice BCSLruleParser), e-cyano API na získanie informácií o jednotlivých objektoch a taktiež jednoduchý algoritmus na rozloženie komponent a generovanie súradníc, ktorý sme vymysleli a neimplementovali, a ktorý je postačujúci pre účely tejto aplikácie.

Syntaktický strom rozdeľuje pravidlo na ľavú a pravú stranu, teda na reaktanty a produkty. Vytvorený algoritmus ako prvé vypočíta vnútornú štruktúru jednotlivých reaktantov a produktov. Postupným prehľadávaním stromu do hĺbky a prechádzaním uzlov, ktoré obsahujú informáciu o agentoch pravidla, sa generujú súradnice pre jednotlivé komponenty pravidla. Základná myšlienka je si vždy uchovávať informáciu o súčasných dostupných súradniciach — *current coordinates*, ktoré indikujú, kde sa ďalšia komponenta môže nachádzať.

Povedzme, že máme pravidlo, ktoré obsahuje reaktant *ps2(chl{*}|p680{n}|pheo{n})*, čo je štruktúrny agent s tromi atomickými agentmi. Ako prvé sa vygenerujú súradnice X a Y určujúce pozíciu ľavého horného rohu pre najväčšiu komponentu (štruktúrneho agenta ps2). Následné sa *current coordinates* posunú o určitú vzdialenosť definovanú globálne pre celé pravidlo, aby nedošlo k tomu, že by sa jednotlivé komponenty neprekrývali a bolo ich od seba možné rozoznať.

V ďalšom kroku sa generujú súradnice pre atomických agentov. Keďže tie už v sebe neobsahujú iné komponenty, tak spolu so súradnicami X a Y sa podľa dĺžky mena agenta vygeneruje výška a šírka SBGN entity, prípadne sa vygenerujú súradnice, výška a šírka taktiež pre EPN stavu atomického agenta, ktorá sa vždy nachádza na hornej hrane atomického agenta. Podobne sa vygenerujú súradnice pre zvyšných dvoch atomických agentov, posúvaním hodnoty *current coordinates* definujúcich voľné súradnice pre ďalšie objekty.

Nakoniec sa určí výška a šírka aj pre štruktúrneho agenta, v závislosti na polohe atomických agentov zanorených v ňom. Rovnaký prístup je aplikovaný aj keď sa jedná o komplexného agenta obsahujúceho ďalšie zanorené EPN agenta do ľubovoľnej hĺbky. Potom čo je takto vypočítaná vnútorná štruktúra jednotlivých reaktantov a produktov sa na základe ich šírky a výšky určí poloha PN procesu a následne prebehne zarovnanie reaktantov a produktov podľa jednotlivých kompartmentov — reaktanty a produkty nachádzajúce sa v tom istom kompartmente sa nachádzajú na rovnakej úrovni a podľa ich finálnych súradníc sa prepočítajú aj súradnice všetkých objektov v nich zanorených.

Následne sa podľa najväčšej X-ovej a Y-ovej súradnice určí výsledná veľkosť celého SBGN diagramu a pomocou SBGN API sa vygeneruje výsledný obrázok. Zo stromovej štruktúry nie je vždy jasné, či ide o komplexného alebo štruktúrneho agenta. Pre určenie tejto informácie sa volá e-cyano API, pomocou ktorého sa určí o akého agenta ide.

5.3.2 Implementácia reakcií

Pri reakciách je to o niečo jednoduchšie. Jednotlivé reaktanty a produkty v sebe nemôžu obsahovať iné zanorené objekty a všetky sú reprezentované jedným SBGN objektom — makromolekulou. Oproti pravidlám navyše zobrazujeme aj modifikátory reakcií, naopak sa vynecháva vizualizácia kompartmentu. Tieto dve skutočnosti pomerne zjednodušujú výpočet súradníc pre komponenty reakcie oproti pravidlám. Reaktanty a produkty sa v diagrame nachádzajú na ľavej a pravej strane a jednotlivé modifikátory sú umiestnené hore a dole. Počet reaktantov alebo produktov určí celkovú výšku a počet modifikátorov definuje výslednú šírku diagramu.

Ako pre reakcie tak aj pre pravidlá platí, že výsledný SBGN súbor vo formáte XML, obsahuje všetky potrebné informácie na vygenerovanie SBGN diagramu vo formáte PNG. Ak vstupný API parameter *as_svg* obsahuje hodnotu *true*, aplikácia prevedie konvertovanie formátu PNG na SVG s využitím nástroja *Potrace* a Python knižnice PIL, ktorá slúži na manipuláciu a úpravu obrázkových súborov. Ako prvé sa pomocou PIL knižnice prevedie PNG súbor do bit-mapového for-

mátu PPM, čo je potrebný medzikrok, aby následne nástroj Potrace, mohol previesť tento súbor do formátu SVG.

V prípade že BCSL reakcia alebo pravidlo bolo reverzibilné, sa pri generovaní SBGN súboru vytvoria dva SBGN diagramy. V prípade reverzibilnej reakcie či pravidla v skutočnosti ide o dve samostatné reakcie či pravidlá a pre lepšiu prehľadnosť je výhodnejšie vygenerovať každé vo vlastnom diagrame. Na výstup sa však vráti len jeden obrázok.

Pomocou Python knižnice numpy, ktorá podobne ako PIL slúži na prácu s obrázkovými súbormi, sa spoja tieto dva obrázky do jedného spoločného a následne je tento súbor vrátený na výstup ako jeden obrázok. V prípade potreby je pred odoslaním konvertovaný do SVG formátu.

Aby aplikácia zbytočne nezaberala veľa miesta na disku, tým že by si ukladala súbory s obrázkami, je každý obrázok, ktorý už nie je potrebný (PNG súbory a PPM súbory z medzikroku pri konvertovaní), sa z disku zmaže. Taktiež sa pri každom volaní API na začiatku volania premaže celá zložka s uloženými obrázkami. Tým sa zabráni zbytočnému zaberania miesta na disku, čo by aplikácia tohoto typu nemala robiť.

5.4 Pohľad do budúcna a možné vylepšenia

V prípade, že v budúcnosti by vznikla potreba, vizualizovať celé modely, nie len jednotlivé reakcie a pravidlá, by bolo potrebné využitie lepšieho algoritmu na generovanie súradníc. Bolo by možné využiť a upraviť niektorý zo známych zložitejších a rozsiahlejších algoritmov pre účely platformy, aby boli schopné pracovať s BCSL modelom. Druhým potencionálnym rozšírením by mohlo byť editovanie a integrovanie niektorého externého nástroja na interaktívne zarovnávanie a prezeranie SBGN diagramov, akým je napríklad SBGNViz. To by však presahovalo koncept a rozsah bakalárskej práce a hodilo by sa ako možná téma na diplomovú prácu.

6 Záver

Po vytvorení návrhu mapovania BCSL do SBGN bolo zistených zopár nedostatkov. Prvým problémom je nedostatočná notácia SBGN pre jazyk BCSL. Najväčší nedostatok SBGN predstavuje absencia EPN, vďaka ktorej by bolo možné rozlíšiť štruktúrneho agenta z BCSL, od komplexného agenta. Možným riešením je rozšírenie štandardu SBGN o potrebnú EPN komponentu, čo vyžaduje spoluprácu s autormi štandardu a návrh grafickej podoby chýbajúcej komponenty, ktorá by bola akceptovaná komunitou, pracujúcou s SBGN.

Druhým nedostatkom je zobrazovanie komplexu v prípade, že model obsahuje komplexnú signatúru a komplex je tým pádom pomenovaný. V tomto prípade aplikácia zobrazuje iba prázdny pomenovaný komplex, bez vnútornej štruktúry komplexu a to z dôvodu absencie informácii o štruktúre komplexu. Možným riešením je rozšírenie e-cyano API o túto informáciu. Ak by došlo k tomuto rozšíreniu, následne by bolo možné štruktúru komplexu pomocou e-cyano API *vyskladať* a zobraziť tak komplex aj s jeho vnútornou štruktúrou.

Samotná aplikácia má tiež obmedzené možnosti. V prípade, že v budúcnosti vznikne potreba zobrazovať celé modely (nie len reakcie či pravidlá), tak budú potrebné rozsiahlejšie zmeny v aplikácii. Napríklad návrh lepšieho algoritmu na rozloženie komponent v diagrame, prípadne pre generovanie súradníc využiť vhodný externý nástroj alebo API.

Na zobrazovanie celých modelov, by sa ale viac hodilo využiť nejaký nástroj, určený práve pre tieto účely. Aplikácia by mohla slúžiť ako medzi krok na spracovanie pravidla a vytvorenie validného SBGN súboru (hoci nie s dokonalým rozložením komponent), keďže väčšina nástrojov na vizualizáciu a automatické rozloženie EPN v SBGN vyžaduje na vstup už validný SBGN súbor vo formáte XML.

Bibliografia

1. HUCKA, M. et al. The Systems Biology Markup Language (SBML): a Medium for Representation and Exchange of Biochemical Network Models. *Bioinformatics*. 2003, roč. 19, s. 524–531. ISSN 1460-2059.
2. NOVÈRE N. H., M.; MI, H.; MOODIE, S.; SCHREIBER et al. The Systems Biology Graphical Notation. *Nature Biotechnology*. 2009, roč. 27, s. 735–741.
3. KLEMENT, M.; ŠAFRÁNEK, D.; DĚD, T.; PEJZNOCH, A.; NEDBAL, L.; STEUER, R.; ČERVENÝ, J.; MÜLLER, S. A Comprehensive Web-based Platform for Domain-specific Biological Models. *Electronic Notes in Theoretical Computer Science*. 2013, roč. 299, s. 61–67. ISSN 1571-0661.
4. TROJÁK, M.; ŠAFRÁNEK, D.; BRIM, L.; ŠALAGOVIČ, J.; ČERVENÝ, J. Executable Biochemical Space for Specification and Analysis of Biochemical Systems. *Electronic Notes in Theoretical Computer Science*. 2018. ISSN 1571-0661.
5. KLEMENT, M.; DĚD, T.; ŠAFRÁNEK, D.; ČERVENÝ, J.; MÜLLER, S.; STEUER, R. Biochemical Space: A Framework for Systemic Annotation of Biological Models. *Electronic Notes in Theoretical Computer Science*. 2014, roč. 306, s. 31–44. ISSN 1571-0661.
6. BREITLING, R. What is systems biology? *Frontiers in Physiology*. 2010, s. 159. ISSN 1664-042X.
7. TAVASSOLY, I.; GOLDFARB, J.; IYENGAR, R. Systems biology primer: the basic methods and approaches. *Essays in Biochemistry*. 2018, roč. 62, s. 487–500.
8. TROJÁK, M.; ŠAFRÁNEK, D.; HRABEC, J.; ŠALAGOVIČ, J.; ROMANOVSKÁ, F.; ČERVENÝ, J. E-Cyanobacterium.org: A Web-Based Platform for Systems Biology of Cyanobacteria. 2016, roč. 9859, s. 316–322.
9. GAWRON, P. et al. MINERVA—a platform for visualization and curation of molecular interaction networks. *Npj Systems Biology And Applications*. 2016, roč. 2, č. 16020. ISSN 2056-7189.

10. GONÇALVES, E.; IERSEL, M.; SAEZ-RODRIGUEZ, J. CySBGN: A Cytoscape plug-in to integrate SBGN maps. *BMC Bioinformatics*. 2013, roč. 14, č. 1, s. 17. ISSN 1471-2105.
11. KRAUSE, F.; SCHULZ, M.; RIPKENS, B.; FLÖTTMANN, M.; KRANTZ, M.; KLIPP, E.; HANDORF, T. Biographer: web-based editing and rendering of SBGN compliant biochemical networks. *Bioinformatics*. 2013, roč. 29, č. 11, s. 1467–1468. ISSN 1460-2059.
12. DOGRUSOZ, U. *VISIBIOweb*. 2018. Dostupné tiež z: `http://bcbi.bilkent.edu.tr/pvs.html`.
13. SARI, M.; BAHCECI, I.; DOGRUSOZ, U.; SUMER, S. O.; AKSOY, B. A.; BABUR, Ö.; DEMIR, E. SBGNViz: A Tool for Visualization and Complexity Management of SBGN Process Description Maps. *PLOS ONE*. 2015, roč. 10, č. 6, s. 1–14.
14. IERSEL, M. P. van et al. Software support for SBGN maps: SBGN-ML and LibSBGN. *Bioinformatics*. 2012, roč. 29, č. 15, s. 2016–2021. ISSN 1460-2059.
15. WIESE, R. *Krayon*. 2018. Dostupné tiež z: `https://github.com/wiese42/krayon4sbgn`.
16. HRABEC, J. *BCSLruleParser*. 2018. Dostupné tiež z: `https://github.com/sybila/BCSLruleParser`.
17. SALINGER, P. *Potrace*. 2018. Verzia 1.15. Dostupné tiež z: `http://potrace.sourceforge.net/`.
18. SMARTBEAR SOFTWARE. *Swagger*. 2018. Verzia 3.0.2. Dostupné tiež z: `https://swagger.io/specification`.