

Použití nástrojů CASE ve vývojářské firmě

Vypracovali:

Farský Jiří
Kytka Roman
Marika Višňová

xfarj06@vse.cz
xtyr01@vse.cz
xvism01@vse.cz

Předmět:
Semestr:

4IT450
letní 2006/2007

Obsah

Obsah	2
1. Úvod	3
1.1 <i>Stručný úvod do světa CASE</i>	3
2. Historie a současnost	3
3. Použití CASE nástrojů při vývoji software	4
3.1 <i>PowerDesigner 12</i>	5
3.2 <i>Gaphor v0.10.4</i>	7
3.3 <i>Enterprise Architect 6.5</i>	7
3.4 <i>Visual Studio 2005</i>	9
3.5 <i>Borland C#Builder 2006</i>	9
3.6 <i>C#Builder 2006 - edice a rozšíření</i>	10
Professional	10
Enterprise	10
Architect	10
3.7 <i>BlueJ</i>	10
3.8 <i>MyEclipse 5.5</i>	11
3.9 <i>Umbrello UML Modeller</i>	12
Verze: 1.5.7 (2007-05-15)	12
4. Přínosy CASE a jejich měření	13
5. Trendy do budoucnosti	13
<i>Seznam obrázků:</i>	14
6. Standardizace	14
6.1 <i>ITIL</i>	15
ITIL –Service Support	15
ITIL –Service Delivery	15
6.2 <i>Příklad rozčlenění skupin produktů</i>	16
6.3 <i>Funkcionalita produktů</i>	16
Elementární funkcionalita	16
Pokročilá funkcionalita	16
Porovnání funkcionalit nástrojů	17
6.4 <i>Porovnání nástrojů</i>	17
6.5 <i>Shrnutí - standardizace</i>	18
7. Závěr	18
Zdroje	19

1. Úvod

Naše práce je rozdělena do tří nezávislých bloků, které spojuje problematika CASE nástrojů. Prvním z nich je malá exkurze do historie CASE nástrojů aneb budeme se snažit zmapovat vývoj CASE nástrojů. V druhém popisujeme nejčastějšího použití, funkcí a charakteristik námi vybraných CASE nástrojů ve vývojářské firmě. Rádi bychom také zachytili směr, kterým se CASE nástroje vydají v budoucnosti. V poslední kapitole (bloku) se zabýváme standardizací (koncepční rámec ITIL) a její svázaností a podporou CASE nástroji.

1.1 Stručný úvod do světa CASE

Jelikož už bylo mnoho napsáno v předchozích seminárních pracích (doporučuji seminární práci - Přehled nástrojů CASE na tuzemském trhu zimní semestr 2005/2006 na www.panrepa.org/CASE, kde je dobře popsána klasifikace CASE nástrojů), chtěli bychom čtenáře naší práce pouze stručně uvést do světa CASE nástrojů.

Informační technologie jsou mimo jiné také světem zkratk, označení CASE není také nic jiného než zkratka pro Computer Aided Software Engineering nebo také Computer Aided Systems Engineering, což v překladu znamená počítačem podporované softwarové (systémové) inženýrství nebo-li vývoj software s využitím počítačové podpory.

CASE nástroje jsou potom nástroje, které primárně umožňují: modelování IT systému pomocí diagramů (člověk lépe chápe obrázek než složitě psané slovo), generování zdrojového kódu z modelu (usnadňuje práci programátorům), zpětné vytvoření modelu podle existujícího zdrojového kódu (reverse engineering), synchronizaci modelu a zdrojového kódu, vytvoření dokumentace z modelu. CASE nástroje jsou postaveny tak, aby podporovaly týmovou práci při vývoji systému, zajišťují sdílení rozpracovaných fragmentů, správu vývoje, sledují konzistenci modelu systému, automatizují některé procesy, hlídají dodržování metodiky, některé umožňují řízení celého životního cyklu aplikací atd, přičemž úspěch využití CASE nástrojů záleží také na vybrané metodice.

2. Historie a současnost

Pokud se chceme zaměřit na historii CASE nástrojů musíme vyjít z názvu CASE konkrétně ze softwarového (systémového) inženýrství.

Definice IEEE 610.12:

„Softwarové inženýrství je aplikace systematického, disciplinovaného, kvantifikovatelného přístupu k vývoji, provozu a údržby softwaru, tj. aplikace inženýrství na software. Také je to studium přístupů dle výše uvedeného.“

Softwarové inženýrství se začalo formovat ke konci 60. let, kdy začala tzv. softwarová krize, kdy výkon hardware předčil vývoj software aneb výkon počítačů přesáhl určitý rozměr, kdy se již nebylo možno spolehnout na tzv. programátorské hvězdy, které vyvíjely

SW. Zcela zlomová byla konference NATO v r. 1968 ve středisku Garmisch Partenkirchen ve Spolkové republice Německo, která popularizovala softwarové inženýrství. Do té doby se počítače využívaly pro vědeckotechnické výpočty, kde záleželo spíše na preciznosti řešení, než na efektivitě výroby. V 70. letech dochází k formulaci základních principů tohoto oboru. V dalších jsou vyvíjeny metodiky (strukturovaná, objeková) pro analýzu. Vzniká také první generace nástrojů pro podporu této disciplíny, což jsou právě CASE nástroje. Mezi první CASE nástroje patřily nástroje, které můžeme nazývat Lower Case, což jsou nástroje pro fázi implementace např. generování kódu – původní myšlenka byla ta, že už nebude zapotřebí programátorů, tzn., že CASE nástroj vygeneruje již samotný kód, který nebude potřeba upravovat a bude se moci tak jak byl vygenerován použít. Což ovšem byla, jak vidíme i dnes myšlenka nesprávná, jelikož existují taková odvětví jako je např. letecké inženýrství, které je velmi specifické a každý rozměr v něm má „jiný kód“. Euforii zažívají CASE nástroje v 80. letech společně s GUI rozhraním,

To byl prvopočátek CASE nadšenosti. Dosud byly dostupné pouze samostatné malé nástroje, nyní bylo možné vše integrovat v univerzálním workbenchu. Strategická příležitost, kterou poskytovala CASE technologie, byla ta, že program může být případně udržován na návrhové úrovni spíše než úrovni zdrojového kódu. Toto by bývalo jedním z východisek z boje programovacích jazyků. Bohužel však nejdříve museli uživatelé CASE nástrojů bojovat se dvěma překvapeními: (a) volba nástroje znamenala zvolení návrhu a metody analýzy. Když organizace nebyla připravena pro toto rozhodnutí, nástroj se stal neuchopitelným (b) naděje, že CASE nástroj zautomatizuje generování kódů, se nenaplnila. Když byla investice do CASE prosazena za podmínky automatického generování kódů, úsilí se poté nevyplatilo. Výsledkem bylo, že dichotomie mezi návrhem a kódem přetrvávala, tzn. když se změnil návrh, musel být změněn i kód. Nebo naopak, když se změnil kód, musel se aktualizovat i návrh, což představovalo ještě větší problém. CASE nástroj, který by mohl být používán jako pouze návrhový nástroj nebo jako zdroj pro všechny druhy nesouvisejících materiálů, se stal brzy příliš drahý, tzn. náklady na učení a údržbu byly příliš vysoké.

Časem, v souvislosti s vývojem strukturovaných a hlavně objektových metod, se problémy a chyby postupně odstranily, tzn. nástroje začaly zajišťovat provázanost jednotlivých modelů, je poskytují možnost zpětného vytvoření modelu ze zdrojového kódu ad.

V současné době existuje nespočet CASE nástrojů, které plní různé funkce, nepomáhají již pouze vývojářům, ale také např. v procesu řízení firmy, kdy se zanalyzují stávající firemní procesy a CASE nástroje potom na základě analýzy vymodelují jak by správně tyto procesy měly fungovat.

3. Použití CASE nástrojů při vývoji software

Jak už bylo napsáno výše - CASE nástroje se již používají v celém životním cyklu vývoje SW. Ať již pro návrh architektury řešení, pak pro samotné generování zdrojových kódů a pak samozřejmě i pro zpětné inženýrství pro získání architektury z již hotových produktů (zdrojových kódů). V této části naší práce se zaměříme na popsání vybraných nástrojů – tzn. popsání jejich funkcí a dalších charakteristik.

3.1 PowerDesigner 12

Výrobce: Sybase

URL: http://www.sybase.cz/buxus/generate_page.php?page_id=110&view=1

PowerDesigner je již tradičně vedoucí hráč v oblasti prostředí pro datovou a objektovou analýzu informačních systémů. Stejně dobře je připraven i pro modelování obchodních procesů. Obsahuje nástroje pro obchodně orientovanou procesní analýzu, která umožní identifikovat klíčová místa a funkce podniku jako takového.

Standardní funkce:

- Pro správu požadavků
- Pro generování dokumentace
- Pro SOA

Je určen pro:

Obchodní modelování

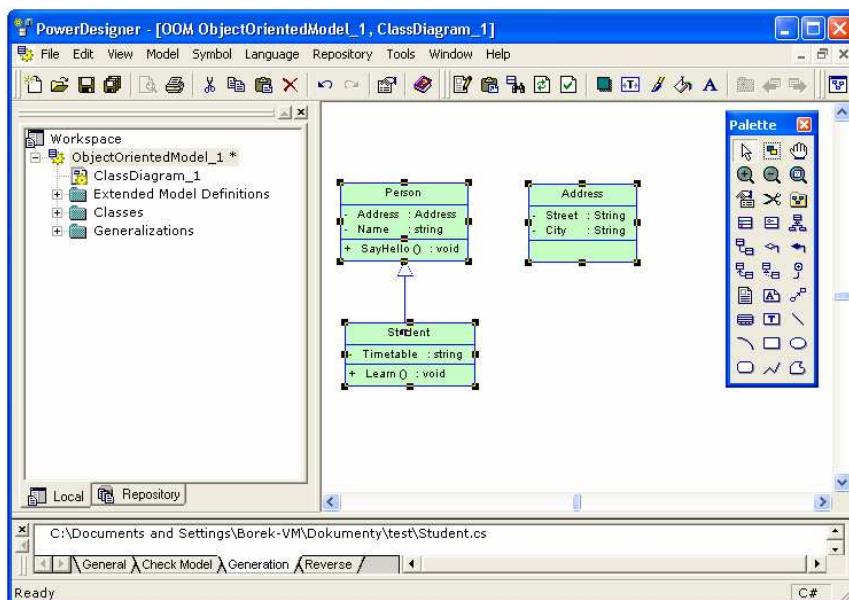
Datové modelování

Objektové modelování

XML modelování

Podporované platformy :

- Process Execution - ebXML, BPEL4WS, podpora servisně orientované architektury
- Relační databáze – plná podpora cca 60 typů databází včetně NCR Teradata
- Objektové jazyky – Java J2EE , C#, VB.NET, PowerBuilder, XML, C++, Web Service
- Integrace s vývojovým prostředím – Eclipse, PowerBuilder a MS Visual Studio



Obr. 1 Screenshot PowerDesigner Zdroj: <http://www.sybase.cz>

Edice PowerDesigneru

PhysicalArchitect

PowerDesigner PhysicalArchitect je základní variantou PowerDesigneru, která obsahuje pouze podporu fyzického datového modelu. PhysicalArchitect je určen především databázovým administrátorům, kteří nepotřebují konceptuální a objektové modely, ale potěší je možnost zpětné analýzy a dokumentace jimi spravované databáze.

DataArchitect

PowerDesigner DataArchitect je ideálním nástrojem pro datového analytika. Nepřekonatelný přístup dvouúrovňového datového modelování dává analytikovi možnost soustředit se na vlastní obchodní data systému a oprostít se od implementačních úprav databáze a přitom vyvíjet databázová schéma přímo na míru cílových databází.

Developer

Pro programátora a návrháře aplikace je určen PowerDesigner Developer. PowerDesigner v této variantě zpřístupňuje ve fyzickém datovém modelu databázové schéma výsledné aplikace a umožňuje pomocí diagramů objektového modelu (Use Case, Sequence, Class, Aktivita, Component) navrhovat příslušnou aplikační logiku systému.

ObjectArchitect

PowerDesigner ObjectArchitect tvoří kompletní sada modulů pro vývoj aplikací. Plně podporuje návrh obou stran informačního systému. Nabízí jak dvouúrovňovou datovou analýzu, tak i návrh aplikace v objektovém modelu.

Business Process Architect

PowerDesigner Business Process Architect umožňuje obchodním (ne-IT) analytikům zachytit procesy v organizaci a předat je IT oddělení k jejich IT analýze a převedení do vyvíjených systémů. Vedle zajištění shody obchodního zadání s vyvíjenou aplikací lze Business Process Architect použít také k popisu obecných procesů v podniku, jejich optimalizaci a pro podporu business process reengineeringu.

Studio

PowerDesigner Studio tvoří univerzální prostředí, které pokrývá veškeré požadavky na modelování a návrh informačního systému. Vedle nástrojů pro detailní návrh informačního systému (konceptuální, fyzický datový a objektový model) obsahuje i model podnikových procesů, který umožňuje analyzovat a modelovat chování celého podniku.

Viewer

PowerDesigner Viewer je určen nikoliv samotným analytikům, ale ostatním uživatelům výsledků jejich práce. PowerDesigner Viewer umožňuje prohlížení všech typů modelů vytvořených v PowerDesigneru a projektoví manažeři pomocí něho mohou sledovat postup práce na systému.

Enterprise Option

Ke každé z výše uvedených variant PowerDesigneru existuje její mutace s podporou centrálního úložiště projektu (repository), která podporuje verzování a řízení práce týmu analytiků.

3.2 Gaphor v0.10.4

Výrobce: open source

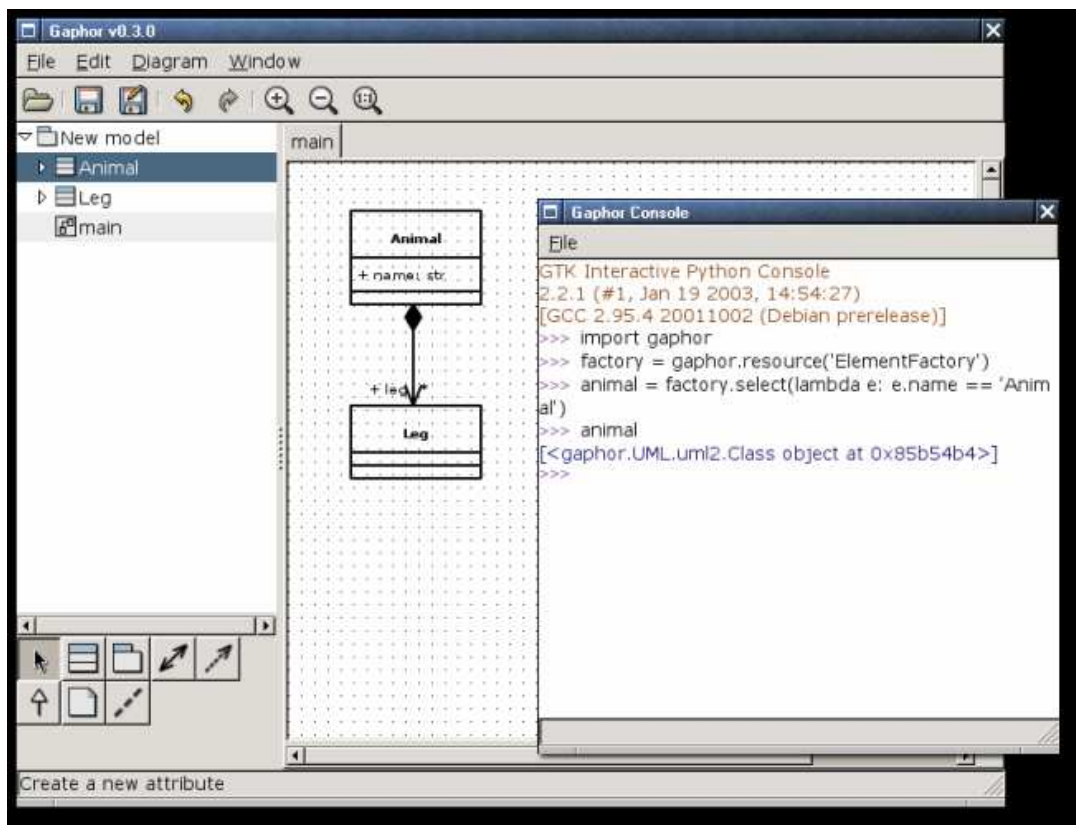
URL: <http://gaphor.devjavu.com/>

OpenSource projekt **Gaphor** je jednoduché návrhové prostředí plně podporující UML 2.0. Již kromě Unixu je dostupný i na platformě Windows.

Podporuje následující typy diagramů:

- Class diagramy
- Use case diagramy
- Action diagramy
- Komponentové diagramy

Podporuje stereotypy pro třídy, rozhraní a balíčky. Umožňuje export do SVG a PNG formátů.



Obr. 2 Screenshot Gaphor Zdroj: <http://gaphor.devjavu.com>

3.3 Enterprise Architect 6.5

URL: <http://www.sparxsystems.com.au>

Enterprise Architect je nástroj pro modelování pomocí UML. EA podporuje následující modely - Business Process Model, Class model, Use Case model, Activity model,

Sequence model a Component model. Výstup je možný ve formátu RTF (dokumentace) a ve formátu XMI pro spolupráci s ostatními produkty. Verze Corporate umožňuje ukládat projekty do databází (MySQL, MS SQL) a obsahuje také podporu pro práci v týmu.

Současná verze EA podporuje tyto modely:

- Model BPM (Business Process Modeling)
- Model tříd (Class model)
- Model případů užití (Use Case model)
- Model aktivit (Activity model)
- Sekvenční model (Sequence model)
- Komponentový model (Component model)

Kromě toho:

- Podporuje řízení týmové práce (eviduje výstupy z projektu, úkoly v projektu apod.)
- Přiřazuje „resource“ k prvkům modelu
- Provádí dokumentaci – výstup generovaný v RTF formátu
- Nabízí výstup modelů v XMI (XMI 1.1) pro kompatibilitu s jinými nástroji
- Zpřístupňuje model pomocí ActiveX interface
- Generuje kód a provádí tzv. reverse engineering do kódu C++, Java, C#, VB.Net, Delphi a Visual Basic.

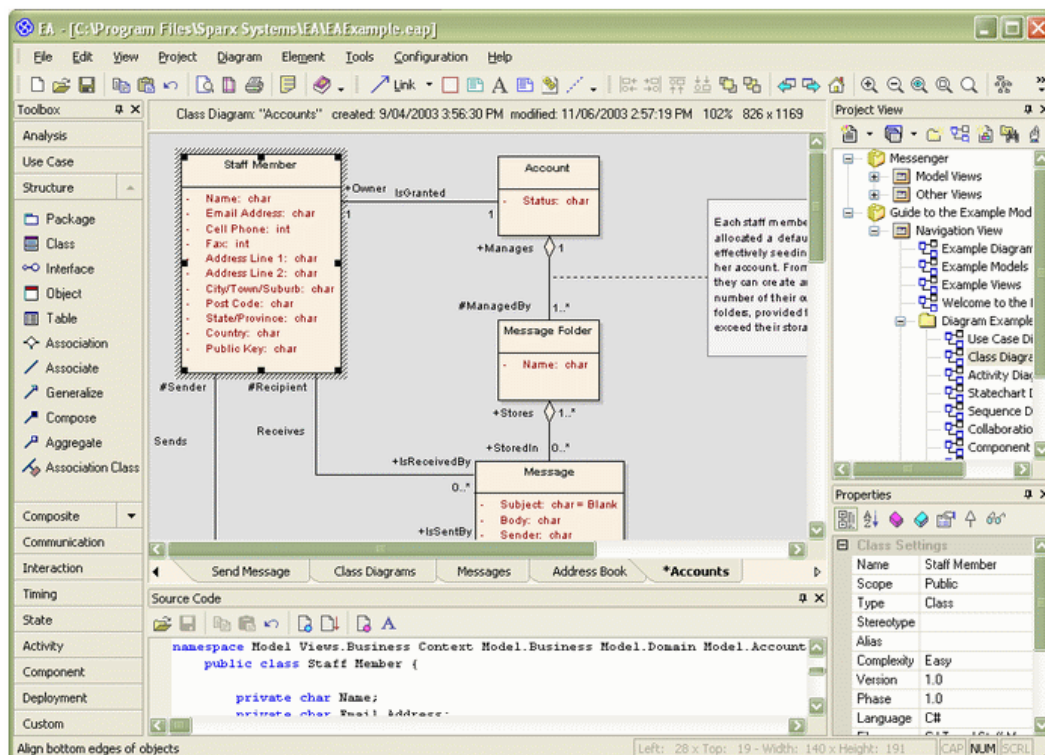
Edice Enterprise Architectu:

Academická licence

Desktop

Professional

Corporate



Obr. 3 Screenshot Enterprise Architect Zdroj: <http://www.sparxsystems.com.au>

3.4 Visual Studio 2005

Výrobce: Microsoft

URL: <http://www.teamsystem.cz/Informace.aspx>

Microsoft Visual Studio 2005 je komplexní vývojové prostředí, které v sobě integruje úplné spektrum nezbytné pro efektivní tvorbu softwaru – od nástrojů pro aplikační modelování až po zátěžové testy.

Kombinuje konzistentní paradigma RAD pro návrh aplikací, intuitivní nástroje pro vývoj serverových aplikací a jednotné prostředí IDE. Visual Studio 2005 obsahuje komponentově orientované vývojářské nástroje a doplňkové technologie, které usnadňují týmovou práci při návrhu, vývoji a implementaci řešení. Pomocí Visual Studia 2005 je možné vyvíjet rozsáhlé aplikace na serverové straně i plnohodnotné aplikace pro malá přenosná zařízení typu PocketPC.

Edice:

Visual Studio Express - odlehčená verze, zdarma

Visual Studio Standard, Professional - verze pro běžné použití jednotlivcem

Visual Studio Team System – podporuje týmovou spolupráci

- Visual Studio Team Edition for Software Architects
- Visual Studio Team Edition for Software Developers
- Visual Studio Team Edition for Software Testers
- Visual Studio 2005 Team Edition for Database Professional

3.5 Borland C#Builder 2006

Výrobce: Borland

URL : <http://www.borland.cz/products/ide.html>

C#Builder 2006 přináší skutečné schopnosti pro podnikový vývoj jako např. modelování UML, řízení životního cyklu aplikací a novátorský produktivní rámec pro .NET. Využívá výhod, které nabízí zabudované objektově-relační mapování, transparentní perzistence objektů a podpora transakcí prostřednictvím rámce ECO (Enterprise Core Objects - základních podnikových objektů). Využívá zpětného inženýrství pro snadnou údržbu aplikací a dvoucestného Live Source™ synchronizujícího kódy a modely. Pomocí integrovaných auditů a metrik zkracuje dobu a snižuje náklady na vývoj podnikových aplikací na platformě .NET, správu požadavků a nástrojů pro správu zdrojového kódu. Zjednodušuje vývoj databází pomocí podpory typu "táhni a pusť", řízení schémat a migrace dat. C#Builder 2006, který je součástí Borland Developer Studia, obsahuje kompletní podporu RAD pro vývoj webových, databázových a GUI aplikací; kromě C# využívá i jazyky C/C++, Delphi a Delphi .NET.

3.6 C#Builder 2006 - edice a rozšíření

Professional

Edice Borland C#Builder 2006 Professional je navržena pro jednotlivce a malé týmy, které se zabývají vývojem aplikací pro PC i webových aplikací s konektivitou na lokální databáze.

Enterprise

Edice Borland C#Builder 2006 Enterprise je navržena pro malé a střední organizace a podnikové vývojové pracovníky, kteří vyvíjejí software podnikatelsky kritického významu s požadavky na vysokou výkonnost databázových serverů.

Architect

Edice Borland C# Builder 2006 Architect je navržena pro týmy odborných vývojových pracovníků, které se potřebují rychle přizpůsobit měnícím se pravidlům podnikání a zvládat důmyslné aplikace, které jsou synchronizovány s vícenásobnými databázovými schémata.

3.7 BlueJ

Výrobce: University of Kent v Canterbury

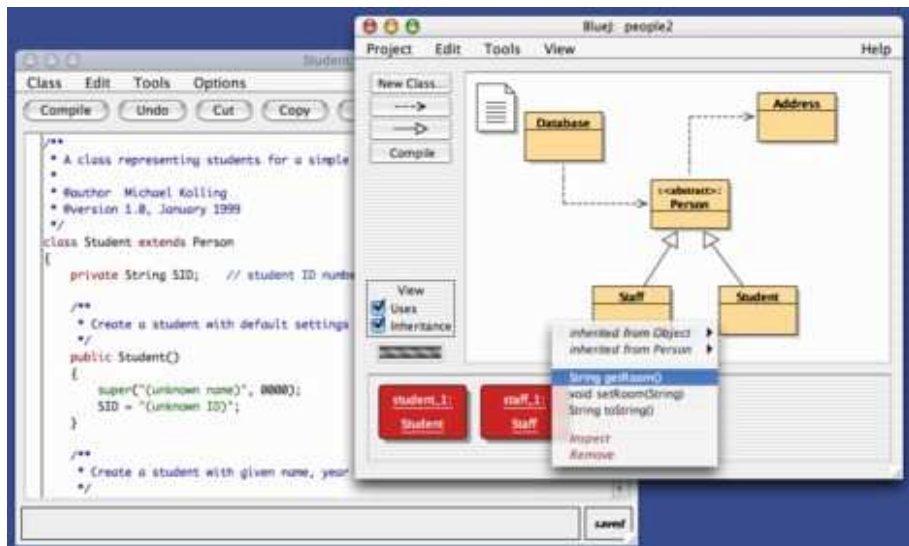
URL: <http://www.bluej.org/>

Verze: 2.1.3 (Windows, MacOS X)

BlueJ byl vyvinut jako součást univerzitního výzkumného projektu pro výuku objektově-orientovaného programování na univerzitě Deakin v Melbourne v Austrálii a University of Kent v Canterbury ve Velké Británii. Tým, který BlueJ vyvíjí chtěl stvořit skučtené objektově orientované prostředí, které bude mít jako základní stavební kameny třídy a objekty, takže jsou nováčci rovnou vedeni k chápání problému objektově. Zároveň chtěli sofistikované prostředí, které však bude jednoduché na použití, tudíž není nutné, aby se začátečníci dlouho učili, jak mají prostředí používat, místo toho, aby se zaměřili rovnou na programování. Prostředí umožňuje interakci a experimentování. Většinu času při výuce zabere strukturování problému do tříd a komunikaci mezi nimi. Začátečníci s tím mají však problém, protože (v jiných IDE) vidí jen řádky kódu či různé komponenty pro tvorbu grafického uživatelského rozhraní. BlueJ ale zobrazuje vizuální zobrazení struktury tříd. Tento nástroj také nabízí určité možnosti ladění kódu pomocí vestavěného debuggeru, který lze snadno ovládat (opět kvůli zjednodušení pro začátečníky). Lze tedy prohlížet vnitřní stav instance objektu, nastavovat breakpointy a krokovat kód. Dále je zde možnost generovat dokumentaci projektu v javadoc formátu

BlueJ poskytuje:

- Integrované IDE (built-in editor, compiler, virtual machine, debugger)
- Editor třídní hierarchie (grafický i textový formát)
- Průvodce pro vytváření objektů
- Interaktivní testování
- Podporu pro inkrementální vývoj



Obr. 4 Screenshot aplikace BlueJ Zdroj: <http://www.bluej.org>

3.8 MyEclipse 5.5

Výrobce:

URL: <http://www.myeclipseide.com>

Verze: 5.5

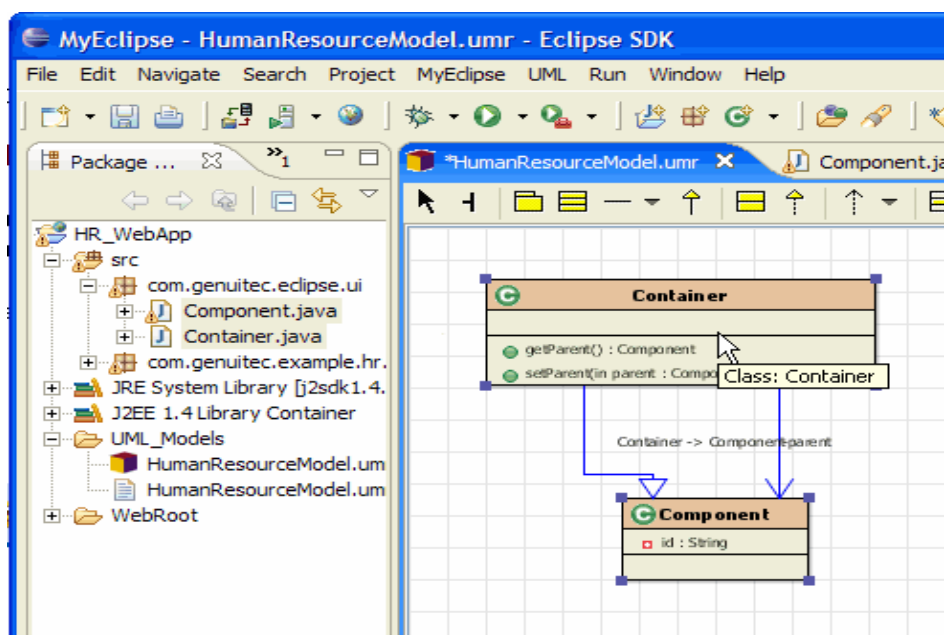
Edice: Standard Edition \$31.75

Professional Edition \$52.95

MyEclipse je kompletní vývojové prostředí pro vývoj v Javě, postavené nad populárním IDE Eclipse. Obsahuje velké množství komponent, které lze samostatně zakoupit. Obsahuje také pokročilý UML modeler, se schopností následně vygenerovat kód v Javě. Podporuje několik typů základních diagramů a exportů do XMI. Umožňuje i tvorbu free diagram, jinými slovy kreslení diagramů bez omezení symbolů i syntaxe.

Zajímavou schopností je i vizuální návrh navigační struktury webových aplikací v Javě, tvořených s pomocí frameworku Struts nebo Java Faces. V tomto režimu se do diagramu vkládají stránky, přechody mezi nimi, a naváží se jednotlivé akce a události, které k nim vedou.

Z obchodního hlediska je zajímavostí fakt, že MyEclipse se běžně neprodává jako krabicový produkt, ale je pronajímán za fixní roční částku, která zahrnuje pravidelné updaty i upgrady. Další zajímavostí je, že 30.3.2007 byl zprovozněn portál pro vývojáře pro Japonský trh. Uživatelé tohoto portálu budou mít přístup a podporu k japonské verzi MyEclipse Enterprise Workbench.



Obr. 5 Screenshot MyEclipse Zdroj: <http://www.myeclipseide.com>

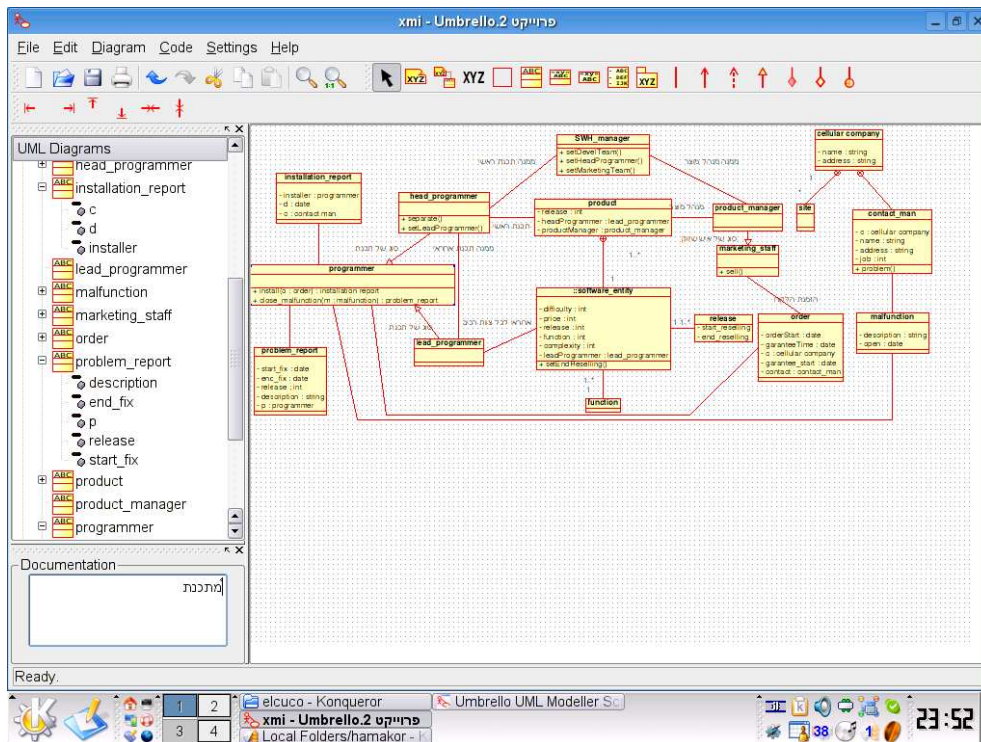
3.9 Umbrello UML Modeller

Výrobce: Opensource

URL: <http://uml.sourceforge.net/index.php>

Verze: 1.5.7 (2007-05-15)

Umbrello je jednoduchý nástroj na vytváření UML diagramů. Pomocí Umbrello se vytváří diagramy softwarových i ostatních systémů ve standardním formátu XMI.



Obr. 6 Screenshot Umbrello Zdroj: <http://uml.sourceforge.net>

4. Přínosy CASE a jejich měření

CASE nástroje jsou hlavně určeny pro snížení náročnosti programování a zvýšení produktivity vývoje softwaru.

Kromě jiných je možné použít následující metriky:

- 1) Zda CASE nástroj umí generovat znovupoužitelné kostry programů a jejich komponent. (zda zdrojový kód vygenerovaný CASE nástrojem se dá použít i v jiném projektu, či je natolik specifický, že pro jiný projekt se bude generovat znovu)
- 2) Porovnávání četností chyb při testování softwarového řešení. (Zda CASE nástroj opravdu přispívá ke snižování chybovosti vyvíjeného softwaru)
- 3) Metrika jak rychle neproškolený vývojář zvládne CASE nástroj ovládat a využít jeho možnosti. (důraz na intuitivní ovládání)

5. Trendy do budoucnosti

Trend vývoje softwaru poháněný obchodem – bude se klást stále větší důraz na obchodní stránku v oblasti IT. Projekty IT budou mít dobře definované cíle a určené návratnosti investic do softwaru.

V současné době, kdy software je zastoupen skoro v každé lidské činnosti (ovládá elektrárny, bankovníctví apod.), je důležité aby byl standardizován celý proces vývoje softwaru. Aby se dalo určit, zda software opravdu dělá to co deklaruje.

Vývojové nástroje budou pronikat do počítačů i obyčejných uživatelů. Bude čím dál tím jednodušší sestavit si vlastní aplikaci, jen pouhým „naklikáním“ („Táhni a pusť“).

Vývojové nástroje budou podporovat širší komunikaci mezi jednotlivými vývojáři i týmy vývojářů.

Budou vznikat nové způsoby úhrad za používání CASE nástrojů. Pod tlakem Open Source komunit musí i výrobci CASE nástrojů hledat cesty, jak zpřístupnit jejich produkty co nejširšímu okruhu zákazníků. Zákazníci požadují větší flexibilitu v cenové politice – např. ceny za upgrade, ceny za používání jen těch modulů, které zákazník opravdu použil, ceny za způsob použití (jiná cena při návrhu architektury a jiná při použití reverse engineeringu) a nebo cena za časové pásmo použití (pronájem na dva měsíce).

Seznam obrázků:

Obr. 1 Screenshot PowerDesigner Zdroj: http://www.sybase.cz	5
Obr. 2 Screenshot Gaphor Zdroj: http://gaphor.devjavu.com	7
Obr. 3 Screenshot Enterprise Architect Zdroj: http://www.sparxsystems.com.au	8
Obr. 4 Screenshot aplikace BlueJ Zdroj: http://www.bluej.org	11
Obr. 5 Screenshot MyEclipse Zdroj: http://www.myeclipseide.com	12
Obr. 6 Screenshot Umbrello Zdroj: http://uml.sourceforge.net	13

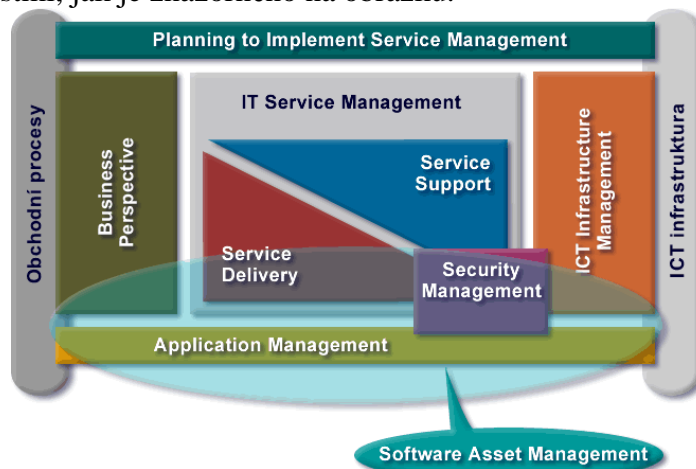
6. Standardizace

V současné době je vyvíjen velký tlak na standardizaci v oblasti CASE nástrojů, zejména v otázce sjednocení s celosvětově uznávanými standardy, jako jsou například ITIL, CobiT a řada dalších. Tyto standardy jsou zastřešovány společnostmi, které je mají ve svém vlastnictví a ty samozřejmě tento tlak na standardizaci také vyvíjejí. Dalším hlediskem vzniklého tlaku na standardizaci je modernost prohlášení, že dané společnosti disponují příslušnými normami a jejich procesy jsou optimalizovány např. dle ITILu atd. Tímto se společnosti prezentují směrem ke svým klientům, snažíc se jim sdělit, že dělají vše pro to, aby vnitropodnikové procesy maximálně podporovaly zákaznickou spokojenost. Na rozdíl od norem ISO 9000 mají tyto standardy jednu obrovskou výhodu: pro splnění ISO je často zapotřebí zavést řadu relativně vágně definovaných kontrolních mechanismů a společnosti ISO zavádějí zejména z důvodu „marketingového a PR“, proto vznikají dokumenty pro dokumenty a procesy pro procesy, nikoliv kvůli vlastní efektivitě. Nechci se tímto nikoho dotknout, ale nesčetněkrát jsem se setkal s tím, že se firma sama přiznala k tomu, že si certifikaci ISO prostě zaplatila a ta byrokracie okolo se nějak obchází. I společnosti, poskytující tuto certifikaci se takto chovají. U standardů typu ITIL a další se mi situace nejeví (zatím?) tak pesimisticky. Společnost může využít koncepčního rámce ITIL pro podporu některých svých procesů, tím získat opravdu reálný přínos pro procesní řízení ve společnosti a již může říkat, že má procesy optimalizované dle ITILu. Samozřejmě, i při této cestě s nejvyšší pravděpodobností naroste na některých místech administrativa, ale obvykle jen proto, že zde opravdu byl nedostatek některého kontrolního mechanismu a vše slouží pro dobro věci. V neposlední řadě tlak na standardizaci roste i proto, že CASE i

další nástroje více a více tyto standardy podporují a jejich uživatelé si na to sami tudíž zvykají.

6.1 ITIL

Když už jsme zmínili koncepční rámec ITIL (pozor! Nikoliv metodiku!), pojďme si na jeho příkladě ukázat svázanost a podporu CASE nástroji. Nejprve jen velmi stručně o ITILu. ITIL je rozsáhlý, konzistentní a procesně orientovaný koncepční rámec pro řízení ICT služeb, založený na nejlepších zkušenostech z praxe. Toliko definice. Sestává celkem z 8 knih (nová verze ITIL v.3 obsahuje ještě některé svazky navíc), které se zabývají jednotlivými oblastmi, jak je znázorněno na obrázku.



Jde o zaměření na podporu interních procesů pomocí ICT technologií. Na rozdíl od ITILu je CobiT zaměřen širěji, ale jeho implementace je zase obtížnější, nemá tak přesně definované procesy a je určen spíše pro vrcholový management, zatímco ITIL je určen pro IT manažery.

V ITILu jsou dvě hlavní knihy Service Delivery a Service Support. Pro naše další povídání si malinko přiblížíme jejich obsah.

ITIL –Service Support

Service Support je zaměřen více na uživatele. Vstupní funkcí (nikoliv procesem), jediným kontaktním místem v ITILu pro zákazníka, je Service Desk. Tudy se dostávají dovnitř požadavky, dotazy atd. Tyto se v ITILu nazývají incidenty, protože obvykle bývají charakteru takového, že je potřeba jejich řešení. Správou incidentů se zabývá Incident Management. Pokud je incidentů více nebo jsou závažnějšího charakteru, mohou vyústit v problém, který je spravován Problem managementem. Problémy mohou následně vyvolat potřebu realizace změny – Change Management – a více změn může vést k vytvoření nové revize, updatu apod. – Release Management. Veškeré atributy předchozích procesů se ukládají za pomoci Configuration Managementu v CMDB (Configuration Management DataBase), kde existuje jejich historie, vývoj atd.

ITIL –Service Delivery

Knihy Service Delivery je zaměřena více na společnost a zabývá se správou a hlídáním nastavení smluv (Service Level Mng.), optimalizací kapacity infrastruktury IT/ICT (Capacity Mng.), dostupností služeb (Availability Mng.), řešením nepřetržitosti dodávání

služeb při kritickém výpadku (IT Service Continuity Mng.) i řízením finančních otázek (Financial Mng. For IT Services).

6.2 Příklad rozčlenění skupin produktů

Produkty, které podporují ITIL, resp. jsou v souladu s ITILEm, si můžeme rozčlenit např. do následujících kategorií:

- Správa incidentů
- Správa změn
- Správa verzí
- Správa prostředků firmy
- Správa úrovně služeb

Pokud vidíte nějakou podobnost s knihami ITILu, tak ta opravdu není náhodná.

6.3 Funkcionalita produktů

Elementární funkcionalita

Podíváme-li se například na Incident Management, můžeme mezi základní funkcionality, nabízené produkty podporujícími ITIL, zahrnout následující:

- Podrobná správa incidentů – jejich identifikace, rozčlenění, naléhavost, řešení...
- Uživatelské role – nastavení rolí a práv k zabezpečení hladkého fungování a zároveň bezpečnosti
- Přiřazení incidentů uživatelům – automatická správa příchozích incidentů a jejich řešení
- Oznamování o vzniklém incidentu – zabezpečení informovanosti a komunikace
- Historie incidentů – záznam do CMDB
- Připojování dodatečných informací (soubory, přílohy, obrázky) – maximální propojení s relevantními dokumenty pro získání maximálního množství informací
- Základní vyhledávání – v historii incidentů a jejich řešení

Pokročilá funkcionalita

Mezi pokročilé funkcionality můžeme zařadit následující:

- Automatické získávání incidentů z monitoringu
- Automatické upozorňování
- Pokročilé statistiky
- Propojování incidentů
- Grafická realizace vztahu incidentů
- Správa problémů
- Přiřazování incidentů k problémům které je způsobují (Problem Management)
- Proaktivní analýza problému metodami AI
- Poloautomatizované řešení incidentu na základě definovaných politik

Zde bychom chtěli zdůraznit zejména Propojování incidentů tak, aby mohl vznikat komplexnější pohled a v případě potřeb realizovat procesy Problem – Change Managementu. Dále Proaktivní analýza problému metodami AI (artificial intelligence,

umělá inteligence), která dává podklad pro vznik expertního systému, učícího se z předchozích zkušeností apod.

Porovnání funkcionalit nástrojů

V následující tabulce je znázorněn seznam nástrojů a funkcionalit, které poskytují. Nástroje zde jsou jak komerční, tak open source, jak ukazuje další tabulka. Funkcionality zde popsané jsou zastoupeny od maximální míry až po pouhé sledování klíčových funkcionalit.

<i>Správa incidentů</i>	HP OpenView	IBM Tivoli	Easy Vista	Assyst	Aegis	AllChange
Automatické získávání incidentů z monitoringu	ANO	ANO	ANO	ANO	NE	NE
Automatické upozorňování	ANO	ANO	ANO	ANO	NE	NE
Pokročilé statistiky	ANO	ANO	ANO	ANO	ANO	ANO
Propojování incidentů	ANO	ANO	ANO	ANO	ANO	ANO
Grafická realizace vztahu incidentů	ANO	NE	NE	ANO	ANO	ANO
Správa problémů	ANO	ANO	ANO	ANO	NE	NE
Přiřazování incidentů k problémům které je způsobují (Problem Management)	ANO	ANO	ANO	ANO	NE	NE
Proaktivní analýza problému metodami AI	ANO	ANO	NE	ANO	NE	NE
Poloautomatizované řešení incidentu na základě definovaných politik	ANO	NE	NE	NE	NE	NE
Napojení na CMDB	ANO	ANO	ANO	ANO	NE	NE
Výpočty nákladů na dané incidenty a Problémy	ANO	ANO	ANO	ANO	ANO	NE
Pokročilé vyhledávání (formou dotaz.jazyků apod.)	ANO	ANO	ANO	ANO	ANO	ANO

6.4 Porovnání nástrojů

V následující tabulce jsou nástroje a některé vybrané atributy k nim. Čerpáno zde bylo ze semestrální práce našich kolegů z jiného předmětu, avšak chybělo zde stanovení například cenového rozmezí, od kdy je cena vysoká apod. Nicméně například v případě IBM Tivoli téměř jistě hovoříme v řádech statisíců.

<i>Správa incidentů</i>	Typ SW	Funkcionality	Cena	Snadnost nasazení	Platformní nezávislost	Možnosti integrace	Shoda s ITILem
HP OpenView ServiceCenter Problem Mng.	KOM	5	VYS	NIZ	VYS	VYS	ANO

IBM Tivoli	KOM	5	VYS	NIZ	VYS	VYS	ANO
EasyVista Service Mng	KOM	4	VYS	NIZ	VYS	VYS	ANO
Axios Systems Assyst	KOM	4	VYS	NIZ	VYS	VYS	ANO
Aegis Service Desk	KOM	2	NIZ	VYS	VYS	NIZ	ANO
AllChange	KOM	3	STR	VYS	VYS	NIZ	Část.
Census	KOM	3	STR	VYS	NIZ	STR	NE
Bugzilla	OPEN	2	FREE	VYS	VYS	VYS	NE
Mantis	OPEN	2	FREE	VYS	VYS	STR	NE
eTicket	OPEN	1	FREE	VYS	VYS	NIZ	NE
H-Inventory	OPEN	1	FREE	VYS	VYS	NIZ	NE

6.5 Shrnutí - standardizace

Závěrem bychom mohli shrnout, že již v současnosti existuje na trhu řada CASE nástrojů, ať již komerčních nebo freewarových, které podporují různé současné standardy. Tato podpora je v odlišných měřácích, od základních funkcionalit až po maximálně pokročilé.

Co se budoucnosti týče, v současnosti vychází nová verze ITILu (v.3), rozšířená o další svazky, částečně se i přibližující CobiTu a dalším metodikám, je zde lepší definování terminologie atd. Stejně tak CobiT již brzy vyjde v další verzi, a to 4.1. Lze očekávat vzájemné přibližování se těchto standardů, lepší návaznosti a lepší a nové možnosti pro podporu CASE nástrojů. Jak tomu ale skutečně bude uvidíme, až tyto standardy vyjdou a výrobci CASE nástrojů na ně začnou patřičně reflektovat.

Obecně však lze říci, že lze očekávat tendenci ke standardizaci a využívání Best practices.

7. Závěr

Myslíme si, že jsme v naší práci poctivě a dle možností zachytili námi stanovené cíle. Historie CASE nástrojů by však sama o sobě určitě vydala na samostnou práci, šlo by o práci rutinní, jelikož k dané tématice neexistuje mnoho materiálů.

O použití CASE nástrojů ve vývojářské firmě - o tom si každá firma rozhoduje sama, samozřejmě pro dnešní robustní IS je jejich využití značnou výhodou, a to nejen kvůli usnadnění práce vývojářům, ale i díky ušetření nákladů a času na vývoj IS. Je škoda, že v naší době ještě nalezneme spoustu společností a i vývojářských firem, které CASE nástroje podceňují.

Zdroje:

<http://objekty.vse.cz/Programovani/Historie-metodiky#CASE1>

http://www.sei.cmu.edu/legacy/case/case_what.html

<http://www.npd-solutions.com/case.html>

<http://www.dbsvet.cz/view.php?cisloclanku=2004052702>

http://www.comphist.org/computing_history/new_page_13.htm

www.lbms.cz

http://www.comphist.org/pdfs/CompHist_9812tla7.pdf

<http://portal.acm.org/citation.cfm?id=1184857.1184987&coll=GUIDE&dl=%23url.coll>

(Tsuda, M.: Effectiveness of an Integrated CASE Tool for Productivity and Quality of Software Developments. Oxford University Press, 2004.)

<http://www.ajaxdevelopersjournal.com/read/47166.htm>

(Brown, Alan W.: i-Technology Viewpoint: The Future of Software Tools)

<http://panrepa.com/>

(Válek O., Páter D., Svoboda R., Markalous D., Pobuda T.: Semestrální práce Nástroje pro vývoj aplikací a jejich vazba na CASE. 2006.)

http://en.wikipedia.org/wiki/ITIL_v3

<http://en.wikipedia.org/wiki/ITIL>

<http://www.itsmf.org/>

<http://www.itil.co.uk/>

<http://www.itil.cz/>

Semestrální práce M. Bergera a M.Špičky

Semestrální práce M.Horkého a R.Kytky

Poskytnuté materiály – knihy Service Support a Service Delivery od Ing. Brucknera

Příspěvek z konference, Bc. Jiří **Skála**, Telefónica O2 Services

Konference ICTM 2007