Vývoj a provoz APV v dnešních podmínkách (požadavky a nástroje)

Ivo Šimek, Christo Kračunov

DELTAX Systems a. s., Na Pankráci 30, 140 00 Praha 4, Česká Republika

Abstrakt

Organizace vždy kladly (nebo by alespoň měly klást) vysoké požadavky na své informační systémy. Chtějí, aby jejich aplikace byly vyvinuty rychle a levně (to platí i o jejich dalším rozvoji), zvládaly stále více kritických funkcí, pracovaly na mnoha místech, ve stejný čas, na různých platformách a rychleji. Navíc očekávají, že funkce jejich aplikací budou spolehlivé, přesné a dostupné, kdykoli to bude potřeba. Nástup technologií a prostředků klient/server a Internetu znamenal významné zvýšení složitosti informačních systémů. Řízení vývoje, nasazení a provozu dnešních rozsáhlejších aplikací je dnes mnohem složitější a náročnější, než tomu bylo dříve v případě systémů centrálního zpracování.

Referát rozebírá podmínky, ve kterých dnes informační systémy fungují, a obecné požadavky, které jsou kladeny na nástroje pro vývoj a testování aplikací a zabezpečení jejich provozu. Dále seznamuje s ucelenou řadou produktů firmy Compuware pokrývající celý životní cyklus aplikace - od nástroje pro vývoj, provoz a údržbu aplikací (*Uniface*), přes nástroje pro jejich testování (*QACenter*), až po nástroje pro řízení, správu a zabezpečení provozu (*EcoTOOLS* a *EcoSCOPE*).

1. PODMÍNKY VÝVOJE, NASAZENÍ A PROVOZU IS

Informační systémy jsou dnes poznamenány především rychlým nástupem architektury klient/server a s tím spojeným rozvojem informačních technologií vůbec. Dalším významným, dnes již nepřehlédnutelným, a v budoucnu zřejmě nejvýznamnějším faktorem utvářejícím podmínky, ve kterých IS vznikají a jsou provozovány, je celosvětový rozvoj využívání služeb Internetu a technologií s tím spjatých.

Dnešní informační systémy se vyznačují několika charakteristikami, které jejich tvorbu a zavádění výrazně komplikují. Jsou to zejména:

Distribuovanost - ve smylu geografickém (části systému jsou rozmístěny v různých lokalitách), funkčním (různé funkce jsou vykonávány na různých místech) a ve smyslu distribuovanosti datové základny (spolu s funkcemi nebo i nezávisle na nich jsou rozmístěna data);

Heterogenita - jsou použity různorodé technické prostředky; operační systémy se liší podle charakteru počítače a podle stavu homogennosti dosažené v systému; komunikační prostředky používají různé protokoly; datové zdroje jsou obsluhovány různými systémy správy (jednotlivé RDBMS, WWW, fulltext, ...); ostatní aplikační vybavení je vytvořeno zpravidla jinými prostředky; architektura systému a aplikací se liší jednak systém od systému, jednak uvnitř systému;

Komplexnost - je nasazeno velké množství různých technologií a ty jsou samy o sobě složité; mezi komponentami v systému existuje velké množství vnitřních a vnějších vazeb;

Dynamika - vývoj je průběžně ovlivňován potřebami hlavního předmětu činnosti organizace resp. podniku; dalším hybným momentem je morální zastarání použitých technologií, architektur, postupů, metod atd.;

Vysoké nároky na provoz - vzhledem ke klíčové roli IS při fungování organizací resp. podniků je nutná trvalá stabilní funkce a vysoký výkon klíčových i podpůrných aplikací; je třeba vyloučit ztráty dat; v současnosti je velmi důležitým aspektem zajištění bezpečnosti systému;

Vnitřní vlivy - je kladen požadavek na plánovitý provoz a rozvoj IS; samozřejmý je rovněž požadavek na minimalizaci provozních a celkových nákladů; stále se zvyšují nároky uživatelů;

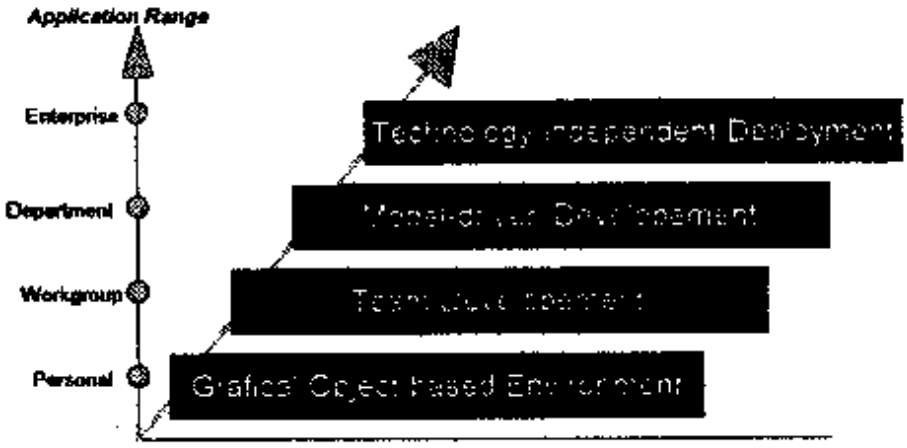
Omezení zdrojů - jsou limitovány všechny zdroje, o kterých lze v souvislosti s IS uvažovat; v oblasti personálních zdrojů je problémem množství a kvalifikace pracovníků v dané lokalitě; finanční zdroje omezují jak vývoj, tak i provoz a rozvoj IS a aplikací; časové zdroje dávají pouze minimum prostoru na řešení provozních problémů a rozvoj IS.

Těmto výše uvedeným skutečnostem se musí tvůrci i provozovatelé IS přizpůsobovat, a to v každé fázi životního cyklu IS. Aby to bylo možné, je třeba mít k dispozici odpovídající technologie. Vedle hardwaru, základního programového vybavení a technologií pro obsluhu dat hrají klíčovou úlohu *nástroje pro vývoj APV a jeho testování a nástroje pro podporu jeho provozu*. Těmito nástroji se budeme zabývat v následujícím textu.

2. NÁSTROJE PRO VÝVOJ APV

Výběr nástroje pro vývoj aplikací je zvláště v případě rozsáhlejších systémů vždy strategickým rozhodnutím, které ovlivní míru úspěšnosti systému nejenom při jeho vývoji samotném, ale i při jeho nasazení, provozu, úpravách a rozšiřování - tedy v průběhu celého životního cyklu aplikace. Je to rozhodnutí, které může ovlivnit úspěch nejenom systému samotného, ale i úspěch organizace, kde je vyvinutý systém vyvíjen respektive nasazen, vůbec.
Každý vyvíjený i již vyvinutý informační systém nutně podléhá tlakům jak ekonomickým, tak i technologickým. Na jedné straně rozhodují *ekonomická hlediska*, která jsou vedle ceny nástroje spojena s produktivitou, a která implikují taková kritéria jako je rychlost a jednoduchost vývoje, nasazení, údržby, rozšiřování a provádění změn v již vyvinuté aplikaci. na straně druhé stojí *hlediska technologická*, která z různých stránek hodnotí nejen technické vlastnosti produktu a to, jak produkt odpovídá současným trendům v oblasti informačních technologií, ale i odolnost výsledku vývoje proti jeho zastarávání. Z těchto hledisek jsou pak vyvozena kritéria odolnosti systému proti změnám technologií (operačních prostředí i architektur informačních systémů obecně) a možnosti využívání technologií nových.

Pro samotný výběr produktu jsou však výše uvedená hlediska příliš obecná, a je potřeba tyto ještě jemněji rozčlenit a přiřadit k nim další. V současnosti je na trhu informačních technologií celá řada nástrojů pro vývoj aplika-cí, které splňují dnes již běžné *základní požadavky* na takovéto produkty - totiž *rychlý vývoj aplikace* (*Rapid Application Development*) *a grafické prostředí* jak při nasazení a provozu aplikace, tak i při jejím vývoji.



Obr. 1. Základní požadavky na nástroje pro vývoj APV

To, co jednotlivé produkty v celé řadě nástrojů pro vývoj aplikačního programového vybavení odlišuje, jsou podle našeho názoru vlastnosti uvedené v následujícím výčtu:

Určení nástroje - zda se jedná o nástroj pro vývoj jednoduchých a malých aplikací pro oddělení a pracovní skupiny pracující v homogenním prostředí na straně jedné, anebo na straně druhé, zda se jedná o rozsáhlé komplexní celopodnikové resp. celoplošné aplikace s využitím grafických a multimediálních informací v heterogenních prostředích;

Součásti produktu - vývojové prostředí, runtime, generátor výstupních sestav, generátor uživatelských ad hoc dotazů, datové konverzní utility, moduly pro definici různých druhů objektů aplikace;

Způsob vývoje - procedurální resp. neprocedurální jazyk, objektová orientace, modelem řízený vývoj s centrálním repository aplikace, využívání předdefinovaných vzorů, možnost vytváření prototypů;

Jednoduchost - požadovaná úroveň teoretických znalostí, nebo znalostí konkrétního operačního prostředí a znalostí o vývojovém nástroji samotném - potřebných ke zvládnutí produktu;

Produktivita - možnost opakovaného využívání již vytvořených objektů aplikace, existence předdefinovaných vzorů, využití definic stávajících datových fondů, možnosti dokumentace eventuelně samodokumentace hotových řešení;

Podpora týmové práce a verzování - přítomnost vlastních nebo převzatých prostředků pro kontrolu oprávnění přístupu a manipulaci s objekty vyvíjené aplikace, definování různých tříd uživatelů nástroje, možnosti pararelního postupu ve vývojových týmech, podpora verzování různých tříd objektů aplikace s prostředky pro vytváření distribučních sad vyvíjené aplikace;

Nezávislost vývoje - uplatnění tříúrovňové architektury s vývojem nad obecným konceptuálním aplikačním modelem - nezávislým na konečném nebo předem vytvořeném fyzickém modelu systému;

Nasazení aplikace v různých architekturách - možnosti provozování vyvinuté aplikace v prostředí různých architektur - aplikační server s terminálovou sítí nebo emulací terminálů (host-based); v prostředí klient/server - pracovní stanice/databázový server (two-tier); v prostředí s pracovními stanicemi a řadou aplikačních a databázových serverů (multi-tier);

Rozložení kódu aplikace - možnosti statické nebo dynamické distribuce kódu aplikace mezi pracovními stanicemi a aplikačními a databázovými servery;

Otevřenost a přenositelnost - podpora různých hardwarových platforem, operačních systémů, prezentačních rozhraní, systémů řízení báze dat a způsob jejich připojení, možnosti práce v heterogenním prostředí a nezávislost na operačním prostředí, možnost přenositelnosti bez potřeby dodatečných úprav;

Návaznost na CASE nástroje - možnost, škála typů přenášených definic a způsob propojení produktu na CASE nástroje, možnosti jednosměrného nebo obousměrného přenosu údajů mezi repository vývojového nástroje a CASE, možnost udržení konzistence a synchronizace údajové základny obou nástrojů;

Uživatelské rozhraní - možnost využití grafických objektů, statické nebo dynamické formuláře upravované koncovým uživatelem, podpora vývoje multilinguálních aplikací, podpora přizpůsobení aplikace různým třídám uživatelů;

Návaznost na operační prostředí - využívání služeb operačního systému, volání 3GL rutin, podpora transakčního zpracování, využití eventuelně suplování vlastností systémů řízení báze dat (databázové triggery, zajištění referenční integrity, volání uložených procedur), vytváření databázových objektů, způsob propojení s databázemi;

Postavení výrobce a dodavatele na trhu - velikost, tradice a renomé výrobce a dodavatele, záruky dlouhodobého postavení a udržení subjektu na trhu, velikost a charakter distribuční sítě, reference a ověření produktu;

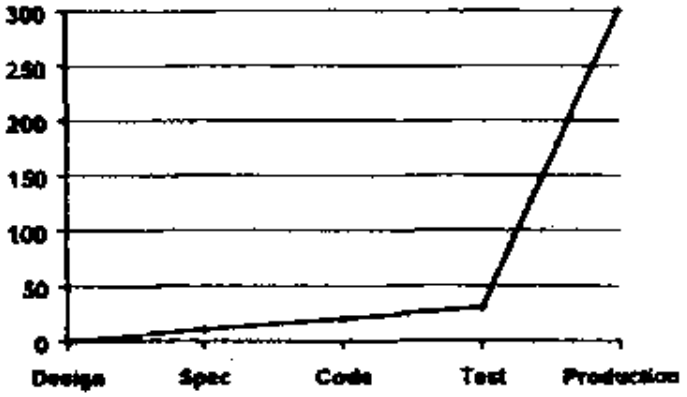
Poskytování dodatečných služeb - technická podpora, školení, konzultace pro různé třídy uživatelů vývojového nástroje, dodávky nových verzí produktu a další služby.

Váhu jednotlivých výše uvedených kritérií při výběru nástrojů pro vývoj aplikací determinuje především charakter, rozsah a složitost zamýšleného IS resp. APV. Míra jejich naplnění pak určuje úspěšnost jak vývoje, tak i nasazení a rozvoje hotové aplikace v celém jejím životním cyklu.

3. NÁSTROJE PRO TESTOVÁNÍ APLIKACÍ

Je zřejmé, že bezchybně navržená a důkladně otestovaná aplikace šetří kapacity, čas a finanční prostředky (odborníci odhadují, že náklady na odhalení chyb v APV činí ročně více než vice než 30 miliard dolarů), které by v opačném případě bylo

potřeba vložit do oprav a zásahů v době nasazení dokončeného systému (viz. Obr. 2.). Nástroje pro automatické testování jsou jednou z cest, jak zajistit spolehlivost a výkon aplikací ještě před jejich finálním nasazením, a tím zajistit i rychlou návratnost prostředků vložených do testovacích technologií i do procesu testování samotného.



Obr. 2 Náklady na odstranění chyb v průběhu životního cyklu aplikace
(pramen: G. Meyers - The Art of Software Testing)

Nástup nových architektur, technologií a prostředků (*klient/server, internet*) znamenal významné zvýšení složitosti informačních systémů. Řízení vývoje a nasazení dnešních rozsáhlejších aplikací je dnes mnohem složitější a náročnější, než tomu bylo dříve v případě systémů centrálního zpracování, i když zajištění správnosti aplikací nebylo nikdy jednoduchou záležitostí. Jestliže *tradiční* aplikace ještě bylo možné testovat *ručně*, uplatnit takový postup v nových podmínkách je velmi těžké a je potřeba najít řešení pro automatické testování.

Při výběru a nasazení nástrojů pro automatické testování musí být posouzeno několik klíčových kritérií:

Podporované kategorie testů - testování je proces, který vyžaduje ověřit funkčnost nejenom jednotlivých programů nebo funkcí, ale i to, jak se tyto jednotky vzájemně ovlivňují, a v neposlední řadě také funkčnost aplikace v podmínkách plného zatížení systému - ať skutečného nebo simulovaného (kategorie testů - viz. Pozn.[1]).

Podporované platformy - je nutné, aby zvolený prostředek podporoval všechny architektury a platformy, které jsou v daném nebo předpokládaném prostředí využívány; každý takový nástroj musí být navíc schopen se přizpůsobovat i nastupujícím trendům v této oblasti;

Přenositelnost - vzhledem k tomu, že většina moderních nástrojů pro vývoj aplikací, umožňuje vývoj na jedné platformě a nasazení aplikace na platformě jiné - je nezbytné, aby spolu s aplikací bylo možné přenášet i testovací nástroje a prostředky;

Způsob tvorby testovacích procedur - testovací procedury (skripty) obsahují příkazy, které simulují fungování aplikace; většina produktů umožňuje vytvářet tyto skripty zachycením aktivity uživatele; navíc dnes mnoho produktů překládá činnost

---

Pozn.[1]: Kategorizace testů

Testy jednotek - testy správnosti izolovaných jednotlivých programů nebo funkcí;
Testy integrity - ověření toho, že izolované jednotky aplikace tvoří jeden funkční celek;
Systémové a zátěžové testy - ověření funkčnosti aplikace v podmínkách plného zatížení systému;
Regresní testování - opakované testování všech nezměněných částí aplikace tak, aby se ověřilo, že provedené změny tyto části negativně neovlivňují.

uživatele do vlastního procedurálního jazyka, který podporuje takové možnosti jako opakování kroků, vyhodnocování podmínek testu, definování a práci s proměnnými, čtení a zapisování z/do souborů, strukturování skriptů do procedur a funkcí; také mohou podporovat funkce specifické pro různé systémy;

Možnosti spouštění testů - interaktivní provádění testů umožňuje uživateli sledovat a interaktivně řídit vykonávání testovacích procedur; mnoho produktů umožňuje krokování testů, definování bodů přerušení, sledování a zasahování do simulovaných vstupních dat anebo do struktury procedury; oproti tomu dávkový způsob spouštění testů umožňuje jejich provádění na pozadí a/nebo v určených časových intervalech - typicky se tento způsob používá pro spouštění rozsáhlých testů;

Způsob verifikace aplikace v grafickém prostředí - zachycení sekvencí stisknutých kláves uživatele a stisků tlačítek myši pro jejich následné využití při automatickém testování je ve většině nástrojů pro testování grafických aplikací obdobné; v čem se jednotlivé produkty liší - je zachycení a popsání pohybu myši respektive ukazatele po obrazovce - pomocí zachycení absolutních souřadnic, anebo zaznamenáváním pohybu mezi jednotlivými pojmenovanými grafickými objekty.

Z hlediska *zajištění a podpory návrhu, řízení a plánování testů* - vzhledem k tomu, že testování komplexní aplikace v prostředí klient/server může běžně zahrnovat i několik tisíc kroků, musí být při výběru nástrojů pro řízení testování posouzena i následující kritéria:

Podpora testovacích metodologií - v současnosti je vyvinuta a implementována řada metodologií strukturovaného testování; většina produktů pro řízení testování je založena na nějakém typu testovací metodologie, obvykle na jednom z průmyslových standardů (ISO, IEE) anebo na vlastní metodologii výrobce;

Řízení z jednoho místa - produkt by měl umožňovat řízení a spouštění všech kroků a testovacích procedur na všech platformách z jednoho místa; jestliže produkt nepodporuje všechny požadované platformy automaticky, měl by poskytovat alespoň možnost manuálních zásahů a spouštění procedur na nepodporovaných platformách;

Integrace s produkty pro automatické testování - každý nástroj pro řízení testů by měl umožňovat automatické spouštění testovacích procedur, nejlépe tedy musí být integrován s nějakým produktem pro automatické testování;

Způsob zachycení chyb - možnost odhalení, zachycení a popsání zjištěných nedostatků je důležitým elementem řízení kvality; takovýto záznam by se měl dát jednoduše udržovat a analyzovat; každý odhalený defekt by měl být na základě předem daných pravidel přidělen k vyřešení zodpovědné osobě; dále by se měl o takto zjištěném a zachyceném nedostatku vést záznam o stavu jeho řešení, dokud problém není odstraněn, a tato skutečnost není ověřena; tento proces by měl být nejlépe opět integrován s použitými nástroji pro řízení a automatické testování.

Produkty pro řízení a automatické testování aplikací v prostředí klient/server - pokud splňují výše uvedené požadavky, významným způsobem napomáhají při odhalování

a odstraňování potenciálních problémů dříve, než tyto mohou nastat. Umožňují urychlit a zkvalitnit proces testování, a tím významně zvyšují efektivitu vývoje a nasazení aplikačního programového vybavení.

4. NÁSTROJE PRO SPRÁVU A PODPORU PROVOZU IS

Od zahraničních analytických firem pochází známé zjištění, že náklady na provoz a údržbu informačního systému tvoří 50 až 70% celkových nákladů za životní cyklus systému. Proto za kvalitním vývojem a zavedením do provozu musí následovat fáze „úspěšného provozování".

Správci informačních systémů jsou postaveni před úkol dosáhnout následujících předpokladů úspěšnosti provozu aplikací:

Kontrola nad provozem - výsledkem dosažení kontroly nad provozem je bezproblémovost provozu a správy aplikací;

Organizování zdrojů informačního systému - výsledkem je výkon a dostupnost aplikací;

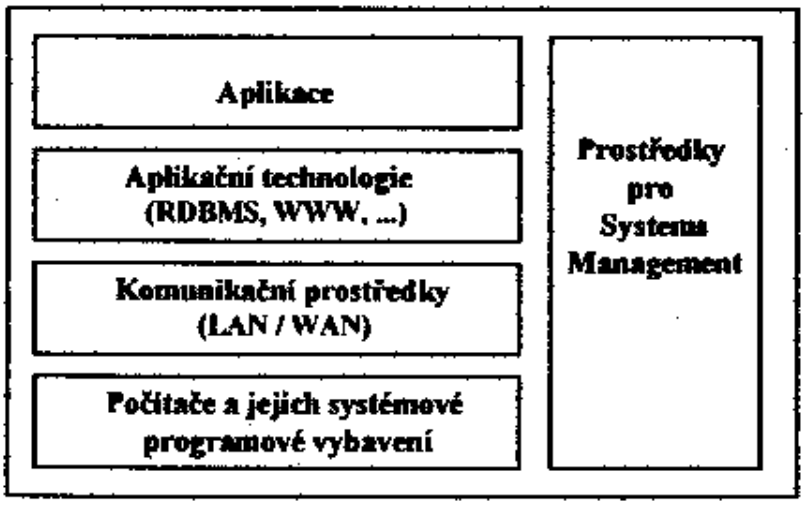
Optimalizace využití zdrojů informačního systému - výsledkem je především minimalizace provozních nákladů;

Ochrana zdrojů informačního systému - výsledkem je integrita a bezpečnost zdrojů.
V informačních systémech současnosti si úspěšné splnění těchto úkolů nelze představit bez nasazení speciálních nástrojů pro správu a podporu provozu IS, tzv. *Systems Management*. Šíře této problematiky je poměrně rozsáhlá. Ve stručnosti uvádíme klasifikaci problematiky v osmi bodech:

Řízení výkonnosti a řešení událostí a mezních stavů - sledování výkonnostních charakteristik systému, jejich vyhodnocování, sledování trendů, prezentaci a ošetřování stavů, kdy se charakteristiky blíží k hodnotám, které jsou nebezpečné pro plynulý provoz;

Správa bezpečnosti - má za cíl ochranu zdrojů, především jejich integrity, řízení přístupu k nim a zajištění utajení a jejich důvěryhodnosti;

Administrace a konfigurace - provádění klasické administrátorské činnosti, jako je nastavování parametrů systému (konta, konfigurace OS nebo RDBMS, ...), reorganizování zdrojů (diskové prostory, periferie, umístění aplikačních balíků a dat) atd.;



Obr. 3 Systems Management v IS

Řízení a provádění technické podpory - řízení a provádění podpory koncových uživatelů aplikací a výpočetní techniky a řízení a zajišťování externích služeb technické podpory dodaných technologií a aplikací v systému;

Řízení zpracování úloh - řízení místa a doby zpracování úloh;

Správa programového vybavení - řízení evidence, distribuce a automatické instalace a deinstalace programového vybavení;

Ekonomika a plánování - podpora finančních analýz poměru nákladů a efektivity vynaložených investic na vybudování informačního systému. Kvalifikované podklady pro plánování dalšího rozvoje informačního systému. (plánování kapacity, vyhodnocování ekonomické náročnosti plánovaných změn v systému atd.).
Výše uvedené úlohy je třeba řešit ve všech logických vrstvách informačního systému. Výsledkem je pak integrovaná struktura znázorněná na obrázku 3.

5. PRODUKTY FIRMY COMPUWARE

Firma Compuware je nadnárodní společnost, která byla založena v roce 1973. Jejím sídlem jsou Spojené státy americké a firma patří mezi 15 největších nezávislých dodavatelů softwarových produktů a souvisejících služeb na světě. Zaměstnává celosvětově více než 3.000 pracovníků.

Compuware se zaměřuje na celou paletu nástrojů pro vývoj, testování a provoz aplikací a nástrojů pro systems management, a to jak pro klient/server, tak i mainframe architektury. Licence k produktům firmy Compuware byly k dnešku uděleny již téměř 9.000 firmám a organizacím na celém světě. Firma si vytkla za cíl poskytovat takové nástroje pro vývoj, testování a provoz aplikací, které by umožnily vyvíjet, testovat a provozovat APV při současné minimalizaci potřebných zdrojů v průběhu celého životního cyklu aplikace. Firma DELTAX Systems, pro kterou pracují autoři referátu, je výhradním distributorem řady produktů této firmy pro Českou republiku a distributorem pro Slovensko.

Obr. 4 Produkty firmy Compuware Inc.pro vývoj a podporu provozu APV

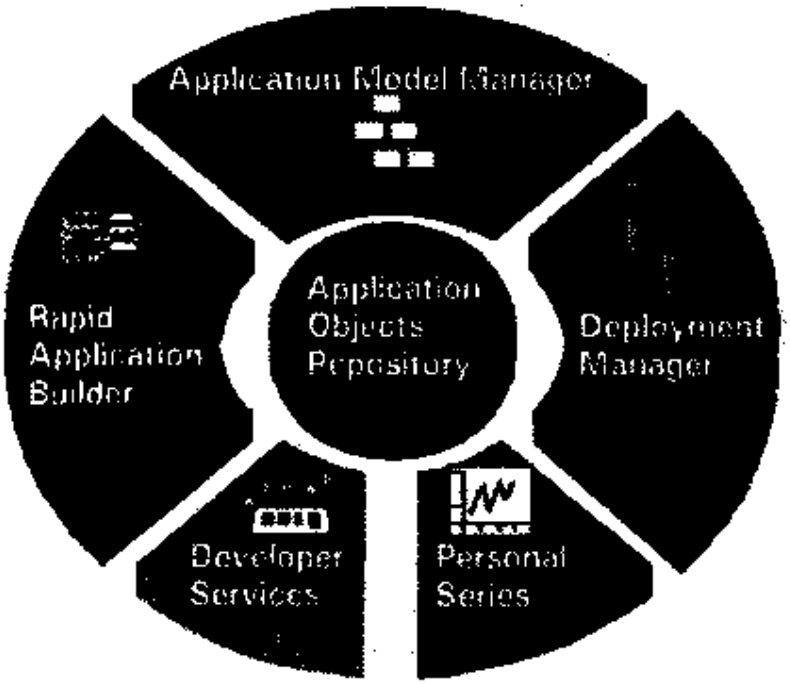
5.1 Nástroj pro vývoj a provoz APV UNIFACE

Uniface je softwarový systém určený pro profesionální tvorbu komplexních aplikací nezávislých na konkrétním implementačním prostředí.

Základní vlastností tohoto produktu je *technologická nezávislost*, a to jak z hlediska provozu v Uniface vyvinutého APV, tak i z hlediska vývoje samotného. Dalším charakteristickým znakem tohoto produktu je *modelem řízený vývoj* - aplikace je

konstruována z objektů aplikačního modelu Uniface a každá změna v modelu se automaticky zpětně promítá do aplikace samotné.

Aplikační model obsahuje definice datových prvků (entity, atributy, vazby) a jejich integritní omezení. Součástí definic jsou i modely chování jednotlivých objektů (t.j. specifikace činností, jež se budou provádět, jestliže se při použití daného objektu vyskytnou předem definované typy událostí) a modely syntaxe a vzhledu objektů. Pro návrh modelu slouží tzv. *Application Model Manager* - grafický nástroj, kde se pro definici chování objektů používá i vlastní 4GL jazyk. Vedle toho lze pro návrh modelu využít i funkce, které umožňují do aplikačního modelu Uniface přenést definice objektů již existujících v databázi. Třetí možností, jak vytvořit aplikační model, je převod definic z repository některého z CASE nástrojů pomocí tzv. *CASE bridge*.



Obr. 5 Komponenty vývojového prostředí Uniface

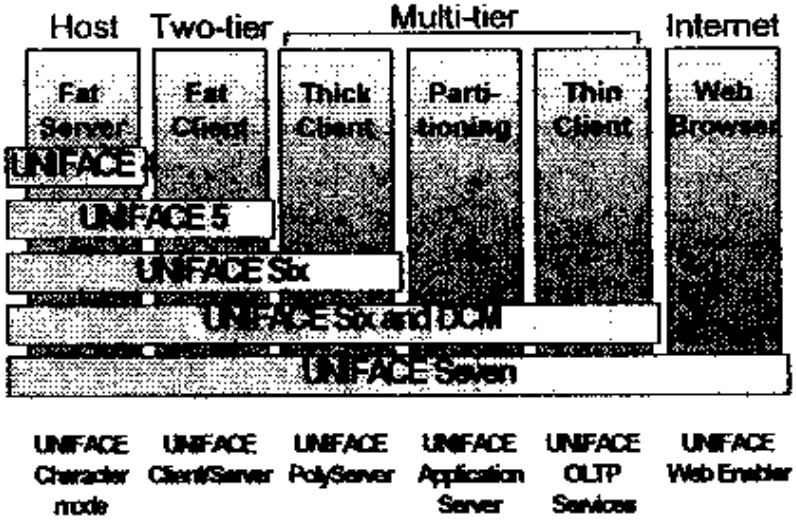
Nad aplikačním modelem je vytvářena aplikace pomocí grafického editoru formulářů (*Rapid Application Builder*). Komponenty aplikace (obrazovkové formuláře, tiskové sestavy, procedury) se zde jednoduchým způsobem konstruují z objektů aplikačního modelu. Pro dosažení plné základní funkcionality komponent aplikace není potřeba psát žádný programový kód. Pro další rozšiřující funkcionalitu má Uniface k dispozici jednoduchý a přitom velmi mocný procedurální 4GL jazyk, jímž je možno definovat chování jednotlivých objektů při výskytu specifikovaných událostí (front-end triggery).

Těžiště návrhu aplikace se tak posouvá více do fáze analýzy, promyšlení věcné podstaty problému a povahy dat a funkcí. Vlastní "programování" se pak redukuje v podstatě na generování aplikací na základě informací uložených v aplikačním modelu.

Výsledkem návrhu aplikace jsou, podobně jako u aplikačního modelu, definice uložené v centrální repository Uniface. Při návrhu aplikačního modelu i komponent aplikace, lze použít předdefinované nebo vlastní předlohy (templates), což vede k zachování standardů a zvyšuje rychlost a produktivitu vývoje, ale i údržby a rozvoje aplikace (mezi komponentou aplikace, vyvinutou na základě předlohy, a danou předlohou je automatický vztah dědičnosti).

Pro účely podpory návrhu a vývoje aplikací Uniface disponuje celou řadou funkcí (*Developer Services*). Jsou to funkce pro *podporu týmové práce* (správa objektů

repository Uniface, řízení přístupových práv členů týmu k objektům repository, komponentám vývojového prostředí a spouštění jejich funkcí a provádění operací). Uniface také disponuje vlastním systémem pro *řízení verzí* (Uniface Version Control Systém). Umožňuje jednak vytvářet historii vývoje objektů aplikace, jednak definovat vývojové verze a ukládat je do speciální části repository. Každý vývojář (člen týmu) si může vedle společných systémových nastavení pro svou práci nastavit ve vývojovém prostředí Uniface i své vlastní *preference*, které jsou nejbližší jeho vlastnímu stylu práce. Pro provádění rozsáhlejších změn v aplikačním modelu, ale i v již vytvořených komponentách aplikace, je možné použít funkce pro tzv. *globální změny*. Pro účely programové dokumentace jsou ve vývojovém prostředí předdefinovány *reporty* nad repository Uniface.



Obr. 6. Architektury pro nasazení aplikací Uniface

Výslednou aplikaci je možné beze změny provozovat plnohodnotným způsobem v různých architekturách (od znakových terminálů, přes dvouřadou i víceřadou klient-server architekturu až po internet - viz. Obr. 6.), na různých výpočetních systémech, nezávisle na použitém grafickém nebo i znakovém uživatelském prezentačním rozhraní s daty uloženými v nejrůznějších typech databázových a souborových systémů. Tím je zaručena přenositelnost a univerzálnost vyvinuté aplikace, integrace dat z různých platforem a využití stávajících datových fondů. Uniface (*Deployment Manager*) k tomu využívá architekturu tzv. driverů, tj. speciálních ovladačů, které pro danou implementaci řídí přístup k různým datovým systémům (relačním databázím, souborovým systémům, mainframe databázím, síťovým protokolům) a interpretují odpovídajícím způsobem příslušné objekty v různých typech zobrazení (znakové prostředí, Windows, Motif). Součástí Deployment manageru jsou i funkce pro vytvoření fyzického modelu dat v cílové databázi (resp. pro vytvoření příslušných scriptů), konverzi dat mezi různými datovými zdroji a vytváření distribučních sad aplikace.

Kromě vlastností prostředí, závisejících na použité technologii, může Uniface při běhu aplikace zohlednit i požadavky vyplývající z *nasazení v různých národních prostředích* a požadavky uživatele vyplývající z jeho profesní specializace. Je možné vytvářet knihovny objektů (menu, informační a chybová hlášení, helpové obrazovky, datumové, měnové formáty, atd.) pro různé národní jazyky. Uživateli se pak tyto objekty zobrazují v tom jazyce, který odpovídá nastavení příslušného parametru v aplikaci (hodnotu tohoto parametru lze aplikaci přiřadit při jejím startu nebo ji i dynamicky měnit za běhu). Stejným způsobem je možné vytvářet podknihovny

(variace) těchto objektů pro různé typy uživatelů (např. programátor, operátorka, vedoucí apod.).

Dalším prvkem, podporujícím přenositelnost aplikací jsou konverzní tabulky. Do repository aplikace je možno ukládat speciální tabulky používané ke *konverzím kódů* při vstupu a výstupu na různá zařízení jako jsou např. klávesnice, tiskárny, displaye a terminály s různým kódováním znakové sady, ale i při čtení a zápisu dat z databází.

V oblasti nasazení aplikace v prostředí *víceřadé klient/server architektury* Uniface poskytuje další vlastní komponenty. Jedná se o *Polyserver* a *Uniface Application Server*.

Polyserver je prostředek, který umožňuje komunikovat připojeným pracovním stanicím bez použití middleware (SQL*Net, Sybase Open Client, InformixNet, ODBC, apod.) zpřístupňujícího ten který databázový server. Polyserver je umístěn na dedikovaném aplikačním nebo přímo na databázovém serveru a komunikace s klientem probíhá v interním Uniface formátu s využitím služeb síťové vrstvy systému (TCP/IP, NetBios, SPX/IPX,...). Některé operace spojené se získáváním dat jsou tak přeneseny z klienta na server (kde jsou předpoklady pro jejich rychlejší zpracování) a je podstatně sníženo zatížení sítě. Polyserver může také fungovat jako distributor zdrojů aplikace (formulářů - programů).

Prostřednictvím *Uniface Application Server* lze synchronně nebo asynchronně spouštět komponenty aplikace (procedury, reporty, dávkové formuláře), které by se jinak vykonávaly na klientovi. Uniface Application server tak slouží jako prostředek pro tzv. partitioning - rozložení logiky aplikace. Různé komponenty aplikace jsou vykonávány tam, kde je to z hlediska výkonu a zatížení nejvýhodnější.

Pro správu systémů, kde je nasazeno více aplikačních a databázových serverů pak slouží *Uniface Name Server*. Tento grafický nástroj umožňuje udržet přehled nad topologií systému a tuto topologii měnit (přemisťovat komponenty aplikace). Uniface Name Server pak při běhu aplikace směřuje požadavky na zpracování podle daného nastavení a umístění zdrojů aplikace.

Pro nasazení v Uniface vyvinuté aplikace v prostředí Internetu slouží *Uniface Web Enabler*. Tato komponenta je prostředníkem mezi WebServerem a plnohodnotnou Uniface aplikací, a - zjednodušeně řečeno, z objektů uživatelského rozhraní generuje HTML dokumenty, čímž umožňuje provozovat aplikace i v prostředí běžně užívaných Web browserů.

Uniface je svým zaměřením nástroj pro vývoj komplexních tzv. business-critical aplikací. Osvědčí se v případech, kdy je třeba používat aplikace v jiných prostředích, než ve kterých byly vytvořeny, kdy jsou části aplikace implementovány v různých prostředích, při rozšiřování stávajících systémů na jiné platformy, integrování starších řešení apod., v případě, kdy je třeba bez přerušení pokračovat ve využívání aplikace při změně hardwaru, databázového systému, atd.. Důležité je, že ve všech těchto případech je možné používat stále tutéž aplikaci. Do oblasti náročné a drahé analytické, designerské a programátorské práce se tedy investuje jen jednou.

5.2 Nástroje pro testování klient/server aplikací

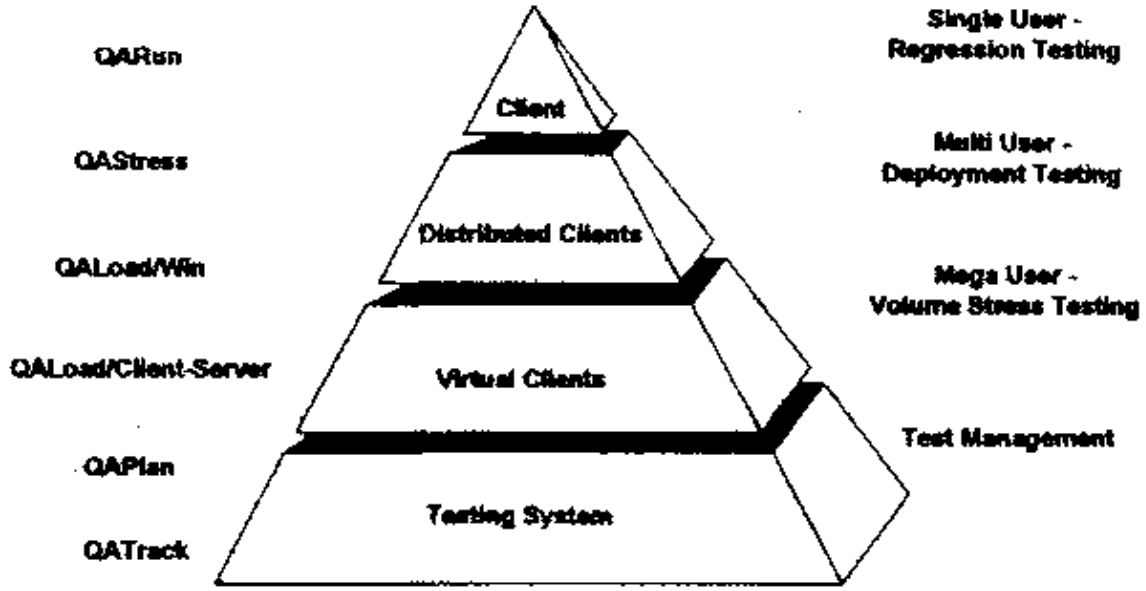
Rodina produktů pro testování klien/server aplikací firmy Compuware *QACenter* tvoří ucelenou komplexní řadu, která zahrnuje nástroje pro všechny kategorie testů APV i pro řízení procesu testování samotného.

Pro *podporu plánování procesu testování* je to produkt *QAPlan*, umožňující návrh a řízení strukturovaného testovacího plánu, alokování zdrojů, návrh testovacích běhů a sledování průběhu prací spojených s testováním. Všechny údaje potřebné pro uvedené činnosti jsou uloženy v repository (m.j. i dokumenty se specifikacemi testovaných funkcí, požadavky uživatelů a změnové protokoly). Pro plánování procesu testování je vždy nezbytné, předem specifikovat potřebné zdroje (technické, personální, atd.) a podmínky, za kterých bude činnost probíhat. Prostřednictvím tohoto nástroje jsou tyto zdroje a podmínky oklasifikovány a je možné popsat jejich vzájemné vazby. Na základě jejich vyhodnocení je pak možné navrhnout testovací procedury (běhy), které popisují manuální nebo automatizovaný proces provedení jednotlivých testů. Tento popis typicky zahrnuje takové údaje, jako jsou čas provedení testu, náklady s testem spojené, požadavky na technické zabezpečení, název procedury, návrh dalšího postupu v případech, kdy test neproběhne úspěšně.

Nástroj *QARun* je určen pro návrh, opakované spouštění a vyhodnocování testovacích skriptů; automatizuje testování aplikací jak v grafickém prostředí, tak i v prostředí znakovém - ve Windows, DOS i aplikací v emulovaném režimu. Zjednodušuje testování tím, že automatizuje opakované, časově náročné činnosti. Pomocí QARun lze vyvinout testovací scripty tak, že akce uživatele jsou transformovány a zachyceny v logickém a přehledném skriptu ve formě příkazů. K tomu nástroj poskytuje mocný jazyk pro návrh testovacích skriptů (zahrnující m.j. příkazy pro větvení, cyklování, vyhodnocování podmínek a definování a používání proměnných). Výsledek průběhu testu lze hodnotit pomocí tzv. kontrolních bodů, které jsou uloženy odděleně od testovacích skriptů. Tyto kontrolní body jsou reprezentovány různými grafickými nebo datovými objekty (soubory, bit-mapy, texty na obrazovce, grafické objekty, atd.), které jsou v průběhu testování porovnávány s aktuálním stavem a na základě tohoto vyhodnocení, lze učinit závěr, zda se testovaná aplikace chová, tak jak se předpokládalo a zda se došlo k očekávaným výsledkům. Nástroj sestává ze dvou komponent - *QARun Command Center* (pro správu, údržbu a archivaci velkého počtu testovacích skriptů, které jsou organizovány do vyšších logických celků) a *QARun Workshop* (zahrnuje editor testovacích skriptů, podporu pro práci s objekty kontrolních bodů, spouštění a zaznamenávání průběhu testu do protokolu, sledování výskytu definovaných událostí v průběhu testu - výskyt chybového hlášení, stisk nějaké kombinace kláves, uplynutí nějakého časového intervalu, apod.).

Na QARun je úzce napojen další produkt pro podporu testování - *QATrack*. Je to nástroj pro zaznamenávání chyb zjištěných během testovacích skriptů, pro sledování jejich odstraňování a celkovou analýzu procesu testování a následného odstraňování zjištěných nedostatků. QATrack v podmínkách vícečlenných testovacích týmů zaznamenává a popisuje v centrální databázi detailní informace o průběhu každého testovacího skriptu (kdo, kdy a jaký skript spustil a jakých dosáhl výsledků). Pro popis průběhu testování využívá data z protokolu (Log) QARun. Tato centrálně uložená data pak nástroj využívá pro sledování výskytu a odstraňování

zjištěných nedostatků (ty lze přiřazovat k řešení jednotlivým členům týmu), a dále umožňuje sledovat kapacitní a časové nároky spojené s testováním. K tomu nástroj poskytuje řadu grafických a textových výstupů (reportů), které dále mohou sloužit pro analýzu problémů vzniklých během testování, vytypování problematických oblastí testované aplikace vůbec a pro podporu rozhodování.



Obr. 7 Nástroje Compuware pro testování klient/server aplikací

Pro zátěžové testy databázových serverů slouží *QALoad*. Tento nástroj umožňuje simulovat přístup velkého počtu klientů k databázovému serveru, a to bez potřeby účasti jak uživatelů - tak i vlastních pracovních stanic. Zátěžový test spočívá ve vytvoření virtuálních uživatelů, a to na určených pracovních stanicích, které jsou řízeny z jedné konzole. Testovací skript, který typicky zahrnuje jeden nebo několik více, či méně složitých SQL příkazů nebo procedur, je na základě zadaných parametrů periodicky či náhodně spouštěn na určených pracovních stanicích a do protokolu se zaznamenávají údaje, potřebné k vyhodnocení výkonnosti databázového servru. QALoad takto získaná data dokáže vyhodnotit - utřídit je do grafických a textových výstupů, které pak mohou sloužit jako podklad pro vyladění databázového serveru a/nebo vyladění aplikace (úpravou SQL příkazů).

Otestovat, jak se nějaká aplikace chová v podmínkách „skutečného" plného zatížení všech zdrojů při práci velkého počtu simultánně pracujících uživatelů, umožňuje produkt *QAStress*. Tento nástroj umožňuje testovacím týmům vyvinout takové testovací procedury, které modelují různé kombinace funkcí systému a počtu uživatelů, a přesně měří chování aplikace a systému v podmínkách různého zatížení. Řídící konzole ovládá daný počet připojených klientů, na kterých podle zadaných parametrů (testovací procedury) spouští buď přímo nějaké aplikace, anebo testovací skripty QARun - to vše buď současně v daném okamžiku, anebo ve specifikovaných časových intervalech. Indikátory selhání, předepsané ve skriptech QARun (například žádná odezva v určeném čase, výskyt chybového hlášení, selhání při porovnání v kontrolních bodech, apod.), jsou zaznamenávány do protokolu. Součástí nástroje jsou opět prostředky pro vyhodnocení údajů získaných v průběhu

testů (počet průběhů jednotlivých skriptů, počet a procento selhání, doby odezvy, apod.).

Klíč k využití výkonu komplexních a rozsáhlých klient/server aplikací spočívá v pečlivém otestování aplikace ještě před jejím uvedením do provozu. Nástroje firmy Compuware reagují na zvýšenou poptávku po takovýchto produktech nabídkou integrované sady nástrojů, které podporují všechny stránky procesu testování.

5.3 Nástroje Systems Management

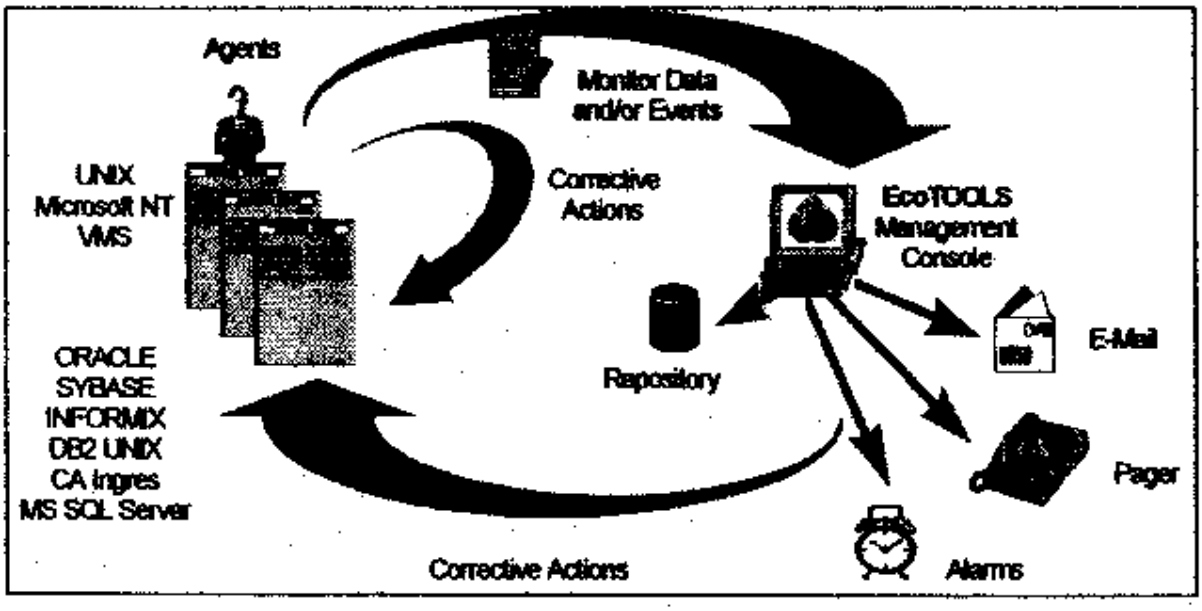
5.3.1 Eco TOOLS

Produkt EcoTOOLS představuje integrovaný nástroj pro správu aplikací v prostředí klient/server. Zasahuje do všech osmi oblastí Systems Management - uvedených v kapitole 4.

Všechny počítače, sítě, databáze a aplikace jsou v případě nasazení produktu EcoTOOLS spravovány z centrální stanice - *řídící konzole*, případně z více kooperujících konzolí. Spravované počítače (stanice, servery) se nazývají *agentské stanice*.

Na provádění širokého spektra operací a funkcí systému EcoTOOLS se podílí jak řídící konzole, tak i agentské stanice. Veškeré řídící a vyhodnocovací funkce jsou centralizovány do řídící konzole. Monitorovací funkce a akční zásahy jsou naopak distribuovány na řízené (spravované) stanice.

Řídící konzole poskytuje administrátorovi kompaktní informace o celém prostředí klient/server. Může automaticky reagovat na podmínky, informovat uživatele a administrátory o určitých stavech nebo v pozadí shromažďovat data. Ta jsou pak použita k analýze, případně jsou z nich vygenerovány přehledné výstupní sestavy. Uživatelský přístup k řídící konzoli je chráněn bezpečnostním subsystémem a kontrolním licenčním mechanismem.



Obr. 8 Schéma funkce nástroje EcoTOOLS

EcoTOOLS monitoruje a řídí komponenty prostředí klient/server prostřednictvím programových agentů. Agent je systémový proces, který vykonává specifické funkce. Ty jsou mu předepsány tzv. scénářem uloženým v řídicí konzoli. V systému EcoTOOLS je připraveno několik stovek agentů a k nim příslušné typové scénáře. Agenti mohou aktivovat další programy nazývané akční, které slouží k provádění opravných akcí reagujících na agentem detekované stavy.

Programové vybavení řídicí konzole EcoTOOLS má kvalitní, objektově orientovaný, grafický uživatelský interface, který umožňuje uživateli jednoduché ovládání a identifikaci událostí a stavů ve spravovaném informačním systému.

Promyšleným nasazením systému EcoTOOLS je možné dosáhnout následujících zlepšení v informačním systému:

Zvýšení dostupnosti aplikací a systému jako celku - detekce problémů dříve než nastanou; automatická inicializace nápravných akcí; automatické uvědomění administrátora.

Snížení nákladů na podporu - předcházení problémům preventivní údržbou, podpora automatických ohlášení problémů; identifikace kritických parametrů; doporučení řešení.

Poskytování expertiz - do funkcionality systému je implementována široká znalostní báze o chování řízených aplikací, která je v případě výskytu nějakého mezního stavu aplikována na proaktivní korekci. V případě, že autonomní funkce není schopna plně ošetřit mezní stav, je poskytnuta přesná specifikace problému a návod k jeho odstranění.

Zajištění kontroly nad prostředím - centralizovaná správa; zachycení historických i právě měřených provozních charakteristik; automatické detekování komponent systému; průběžné sledování bezpečnostního stavu informačního systému.

Optimalizace výkonu aplikací - korelace vzájemných souvislostí; sledování využití zdrojů; přesné označení příčiny vzájemné potřeby zdrojů; identifikace zdroje (nej)náročnějších SQL příkazů.

Zlepšení plánování systémů - shromažďování historických dat; identifikace úzkých míst ve využití zdrojů; poskytování informací pro účtování využití zdrojů.

Produkt EcoTOOLS podporuje unixové systémy a prostředí Windows NT. Z databázových systémů to jsou Oracle, Sybase a Informix.

5.3.3 EcoSCOPE

Produkt EcoSCOPE je určen pro sledování provozu aplikací v sítích LAN a WAN. Identifikuje aplikace komunikující po síti, zjišťuje množství přenesených dat v rámci těchto aplikací, použitý síťový protokol, množství přenesených dat mezi klienty, mezi klienty a směrovači a mezi klienty a servery, doby odezvy aplikací.

Produkt sestává ze tří komponent:

Single View představuje měřicí konzoli, určenou pro sběr informací od měřicích agentů, interpretaci těchto dat a jejich zobrazování do tzv. topologické mapy, ve které jsou zobrazeny aktivní prvky sítě, které mezi sebou komunikují. Komunikace je graficky znázorněna spojnicemi, jejichž tloušťka představuje intenzitu toku dat. Umožňuje získat rychlý přehled o zatížení aktivních prvků sítě datovým přenosem, o nejvýznamnějších aplikacích a použitých protokolech. Významnou vlastností je, že Single View dokáže exportovat data ve formě importovatelné do jiných produktů a do komponenty Report Navigátor. Ten pak, mimo jiné, umožňuje předání dat ve formátu ACCESS a Excel. Single View zobrazuje všechny síťové komponenty, sledovaný provoz v počítačové síti, zatížení sítě aplikacemi, odezvu komunikace klient/server.

Další komponentou je tzv. měřicí agent *Super Monitor*, který slouží k periodickému sběru a vyhodnocování informací získávaných analýzou hlaviček všech paketů vyskytujících se ve stejné kolizní doméně. Super Monitor provádí identifikaci komunikací známých aplikací definovaných v jeho knihovně (předdefinováno je 1200 aplikací, jako např. Oracle, Sybase, Informix a ostatní RDBMS, WWW, Lotus Notes, ftp, X-Window včetně několika desítek nejrozšířenějších počítačových her a podobně) a předává tyto informace měřicí konzoli. Umožňuje rovněž dodefinovat příznaky libovolné aplikace pro její automatickou identifikaci. Je schopen iniciovat varování při některých událostech, jako například překročení definované doby odezvy aplikace a podobně.

Třetí komponentou je databázový prostředek *Report Navigátor* pro analýzu dat získaných z prostředí Single View. Tento prostředek umožňuje analyzovat naměřená data z hlediska odkud, kam, jaké množství dat, mezi klientskými aplikacemi či směrovači bylo přeneseno a v jakou dobu a na jakém protokolu tato komunikace probíhala. Výstupem jsou protokoly v tabelární a grafické formě s tím, že uživatel systému ECOSCOPE má možnost přenést provádění analýzy on-line do produktu MS-EXCEL.

EcoSCOPE indikuje všechny aktivní prvky počítačové sítě jako jsou: routery, bridge, servery, klientské stanice a přepínače. EcoSCOPE komunikuje se směrovači, servery, síťovými kartami, přepínači na základě podporovaných síťových protokolů.

EcoSCOPE je určen pro MS Windows 95 a NT. Podporované protokoly jsou TCP/IP, NetBIOS, AppleTalk, XNS, IPX/SPX, SNA, DecNet, Banyan IP.

6. ZÁVĚR

Vyvíjet a provozovat aplikační programové vybavení v dnešních podmínkách, tak jak jsou specifikovány v úvodu tohoto příspěvku, rozhodně není jednoduchou záležitostí. K tomu, aby nástroje, které by tento proces měly usnadnit, splnily své poslání, musí odpovídat mnoha různorodým kritériím a nárokům, které jsme se na této malé ploše pokusili alespoň ve zkratce naznačit. Firma Compuware (pravděpodobně jako jediná) nabízí ucelenou řadu nástrojů pro vývoj, testování a provoz aplikací (nejenom) v prostředí klient/server.