Dokumendikesksed tarkvaratehnoloogiad

Liitdokumendid (*Compound Documents*)

"Liitdokumendi tehnoloogia on tegelikult komponenttarkvara. Ta lubab luua oma rakendusi lihtsalt paigutades komponente standartsele dokumendile, mis oskab neid hallata."
David Linthicum

Dokumendi kujund lubab intuitiivselt grupeerida seotud objekte, kujutada neid sujuvalt aknas, salvestada neid jagatud faili ja saata neid arvutivõrgu vahendusel teistele töölaudadele (*desktops*) ning serveritesse. Lisaks säilitab dokument püsivaid klient/server sidemeid, mis lubab neil uputatud{manus-} (*embedded*) komponentidel saada andmeid servereist igalpool ettevõttes. Seega saab dokument universaalseks kliendiks serveritele. Tegelikult võivad serverid saata oma klientidele otse liitdokumente, et luua liideseid nende pakutud teenustele -- laiendatud variant WWW'st (WWW + Java ?).

Liitdokument pole midagi muud, kui komponentide hulga organiseerimise kujund, nii visuaalselt kui ka läbi sisalduvussuhete.

Liitdokumendid on kohaks, kus elavad komponendid, isegi töölaud on kujunemas suureks liitdokumendiks, mis integreerib sujuvalt rakenduste ja operatsioonisüsteemi(de) teenused.

Liitdokument on eelkõige visuaalne konteiner. Kasutaja on võimeline looma rakendusi, lohistades (*dragging*) komponente palettidest konteineritesse. Kuna ka konteinerid ise on samuti komponendid, võib neid samuti teiste komponentide sisse paigutada, seega on kõik väga rekursiivne.

Andmed liitdokumendis võivad olla mistahes tüüpi, kuna andmeid haldavad komponendid, kelle omad andmed on. Samuti, kui ilmneb vajadus lisada dokumenti uut tüüpi andmeid, võib seda teha ilma rakendust muutmata.

Ümberlülitused visuaalsete komponentide vahel dokumendis on vähemhäirivad, kui klassilkaliste rakenduste vahel -- igat komponenti saab paigapeal muuta ilma mingi lisaprogrammi käivitamiseta. Paigapeal redigeerimine võimaldab komponendil oma redigeerimisvahendid (menüüd, tööriistaribad jms.) konteinerile lisada.

Liitdokumendid teevad väljatöötajatel lihtsamaks eraldi väljatöötatud komponentide salvestamiseja kasutajatele lihtsamaks dokumentide vahetamise. Andmed dokumentidesse võib koguda erinevaist andmelätteist kasutades klient/server tehnoloogiat.

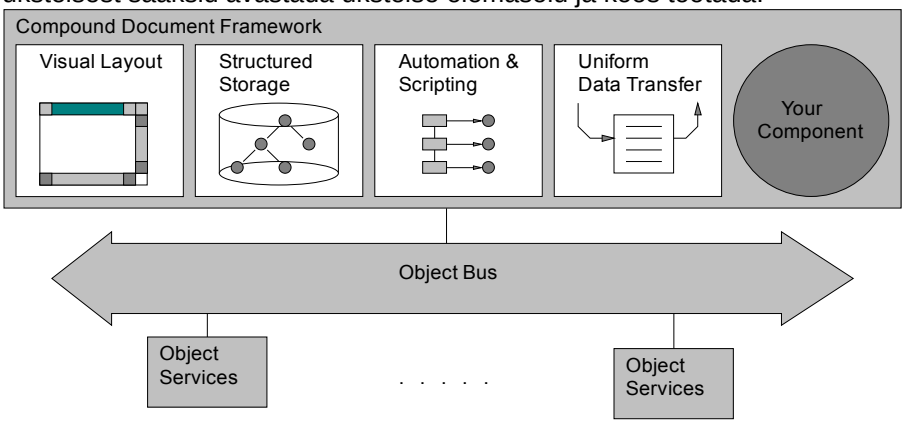
Liitdokumente saab kasutada töö marsruutimiseks ühest masinast (ühelt töökohalt) teise. Marsruuditav dokument võib sisaldada töö all olevat materiali, kasutajaliidese elemente ja stsenaariumikirjeldusi (*scripts*) mis muudavad dokumendi intelligentseks.

Liitdokumentide tehnoloogia lubab seega visualiseerida erinevate komponentide koostööd, esitades seda visuaalselt dokumendina. Igasugused visuaalsed metaforid on komponentide vaheliste suhete esitamisel väga kasulikud (eriti sisalduvuse esitamisel). Parimad komponentide (objektide) kujundid on igapäevaste meid ümbritsevate objektide abstraktsioonid (klassikaline prügikast!). Graafilised töölauad võivad hästi simuleerida igapäevaseid asju -- kasutaja suudab hästi komponente ära tunda ja neid kasutada. Nähtamatutele/kompimatutele asjadele on aga samas raske häid kujundeid leida.

Liitdokumendi raamistik (*Component Framework*)

> *"Kogu liitdokumendi idee on võimalus segamini kasutada erinevaid sisusid ja luua piirid, mis näitavad kus üht sorti sisu lõpeb ja teine algab. Seda kõike tuleb teha, ilma et osad kaotaks oma identiteeti või piire."*
>
> -- Kurt Piersol, OpenDoc'i Arhitekt

Liitdokumendi raamistik annab protokollid, mis lubavad dokumenti tervikuna haldaval komponendil suhelda dokumenti moodustavate komponentidega. Need protokollid lubavad antud komponentidel samuti jagada selliseid ressursse, nagu dokumendi fail või ekraanil paiknevad aknad. Tähtis on, et need protokollid oleksid küllalt üldised, et sõltumatult väljatöötatud komponendid, ilma eelteadmisteta üksteisest saaksid avastada üksteise olemasolu ja koos töötada.



Dokumendi laotuse (paigutuse) teenus (*Document Layout Service*) määrab reeglid, mille alusel iseseisvalt väljatöötatud komponendid jagavad konteineri akent. Komponendid suhtlevad, kasutades konteinerit, kui vahendajat luues kasutaja joks ühtse dokumendi. Konteinerid aktiveerivad komponente, mis paiknevad dokumendis ja eraldavad igaühe jaoks tüki vaadeldavat pinda. Komponendid kuvavad oma andmeid neile eraldatud pinnal ja suhtlevad kasutajaga.

Konteinerid jaotavad sündmusi oma komponentidele ja teatavad, kui midagi neid huvitavat juhtub nende läheduses. Samuti korraldab konteiner jagatud ressursside nagu menüüriba kasutamist. Konteiner peab suutma oma piiridesse kukutatud komponendi vastu võtma ja aru saama, mis sellega teha tuleb.

Ühtne akendesüsteem ühes aknas?

Struktuurne mälu (*Structured Storage*). Tradistsiooniline dokument on monoliitne andmeblokk (tavaliselt failis) mida juhib üks rakendus. Vastupidiselt sellele koosneb liitdokument mitmetest väiksematest andmeblokkidest, kus igat blokki juhib vastav tarkvarakomponent. Liitdokument pakub protokolle, mis takistavad komponentidel lõhkuda ühist dokumenti, milles nad elavad. Liitdokument peab olema võimeline jagada faili mäluühikuteks, mida saab eraldada iseseisvatele komponentidele. Endale eraldatud mälusse võib iga komponent salvestada mida ta tahab.

Kui liitdokument avatakse, otsib ta üles ja aktiveerib komponendid, mis manipuleerivad andmetega. Konteiner ei kasuta ega muuda kunagi andmeid, mis kuuluvad tema sees olevatele komponentidele.

Struktuurne mälu on tehnoloogia, mis loob "failis failisüsteemi", kus igale komponendile antakse sisukorra taoline struktuur oma andmete organiseerimiseks ja kirjeldamiseks, kusjuures andmed ise salvestatakse voogudesse (*streams*).

Andmed tulevad liitdokumenti mitmetest lätetest, s.h. võõrastest komponentidest ning SQL andmebaasidest. Konteiner aktiveerib võõrad komponendid kas säilitades võõraid andmeid otse dokumendis või säilitades viitasid või sidemeid (*links*) välistele andmelätetele.

Ühtne failisüsteem ühes failis?

Stsenaariumid ja automatiseerimine (*Scripting and Automation*) on tähtsamaid liitdokumendi komponentmudeli omadusi. Stsenaariumid lubavad kasutajatel oma rakendusi järelhäälestada. Stsenaariumid lubavad programmeerijatel ja süsteemide integreerijatel luua klient/server suhteid, mis kasutavad dokumendi kujundit ja n.ö. intelligentseid dokumente.

Stsenaariumid võivad näiteks:

- dokumendil pidada arvet lugejate ja muutjate üle
- teatada omanikule e-posti kaudu enda kasutamisest

- vastavalt kasutaja õigustele juhtida mida kuvatakse
- küsida paroole ja valideerida digitaalseid allkirju
- dünaamiliselt pärida dokumenti andmeid (näiteks andmebaasist)
- ...

Antud tehnoloogiat saab kasutada samuti koostamaks liikuvaid komponente, mis roomavad võrkudes ja teostavad mingeid tegevusi.

Ühtne andmeedastus (*Uniform Data Transfer*). Liitdokumendid peavad olema võimelised vahetama andmeid oma ümbrusega. Ühtne andmeedastus pakub ühte andmevateuse mehhanismi, mis on kasutatav mitmete protokollidega -- s.h. lõgetelaua lõika-kleebi (*clipboard cut-and-paste*), otsese manipuleerimise "nihuta-kukuta" (*drag-and-drop*) ja sidumine (*linking*). See andmevahetusmehhanism peab olema võimeline edastama tavalisi andmeid ja terveid komponente nendes sisalduvate komponentidega.

Kokkuvõttes võib liitdokument sisaldada komponendi seisundit, andmeid, intelligentsi ja kasutaja liidest. Liitdokumendid pakuvad raamistikku ja visuaalset kujundit komponentide organiseerimiseks. Nad pakuvad protokolle, mille läbi sisalduvad komponendid liidavd oma kasutajaliidese elemendid konteineri aknaruumi ja jagavad dokumendi faili. Töölaua operatsioonisüsteemid ise on muutumas liitdokumentide raamistikeks, tekitades komponentidele massilise turu.

Vaatleme **OLE**'d ja **OpenDoc**'i, mis on kaks *de facto* standardit liitdokumentidele. Liitdokumentide raamistikud sisalduvad ka NeXT'i *OpenStep*'is, Taligent'i *CommonPoint*'is ja Integrated Objects'i *Newi*'s.

OpenDoc'i Liitdokumendi mudel

OpenDoc on komponenttarkvara arhitektuur, mis on realiseeritud kui hulk platformist sõltumatuid klasse ja teenuseid.

Komponente nimetatakse OpenDoc'is osadeks (*parts*). Osad paiknevad liitdokumendis ja nad ei ole võimelised liitdokumendist eraldi eksisteerima. OpenDoc'i osa koosneb liitdokumenti salvestatud andmetest ja osaredaktorist (*part editor*), mis manipuleerib tema andmetega. OpenDoc'i osa pole midagi muud, kui CORBA objekt, millel on töölaua lisad. Seega laiendab OpenDoc CORBA ORB töölauale. Osad töölaual võivad kasutada CORBA ORB'd koostööks teiste töölaua osadega ja server-objektidega. ORB pakub sealjuures läbipaistvat (*transparent*) juurdepääsu hajuskomponentidele ja CORBA teenuseid nagu turvateenus, transaktsioonid ja nimeteenus.

OpenDoc kasutab tuntud dokumendi kujundit organiseerimaks komponente töölaual visuaalselt.

OpenDoc määrab osad suhtlemise reeglid, et:

sujuvalt jagada ekraanipinda aknas;
salvestada andmeid ühte konteineri faili;
vahetada informatsiooni teiste osadega sidemete, lõigetelaua ja otsese manipuleerimise kaudu;
koordineeride oma tegevusi stsenaariumite, semantiliste sündmuste ja CORBA meetodite väljakutsete abil;
töötada koos teiste töölaua komponentide mudelitega (OLE, Taligent, ...).

OpenDoc koosneb:

- CORBA'le vastavast objektisiinist SOM (IBM'ilt)
- Struktuursest mälust Bento (Apple'ilt)
- Ühtsest andmevahetusest (*Uniform Data Transfer*) (Apple'ilt)
- Liitdokumendi haldamisest (*Compound Document Management*) (Apple'ilt)
- Avatud stsenaariumiarhitektuurist (*Open Scripting Architecture -- OSA*) (Apple'ilt)

OpenDoc'i ei pruugi vastandada OLE'le, teda võib vaadelda, kui abstraktsioonitaset OLE peal. Novell'i poolt väljatöötatud *ComponentGlue* pakub võimaluse ühendada OpenDoc'i ja OLE'd. See tähendab, et OpenDoc'i konteiner näeb OLE objekti kui OpenDoc'i osa ja vastupidi.

SOM (System Object Model)

SOM on OpenDoc'i osade (komponentide) lokaalse ja kauge kooskasutuse aluseks. SOM on keelest sõltumatu, CORBA standardile vastav ORB, mis lubab objektidel suhelda samas aadressruumis või erinevates aadressruumides kas samas masinas või üle võrgu. Tänu sellele võib osade väljatöötaja kasutada suvalist teenust suvalisel CORBA'le vastaval ORB'l.

SOM on samuti komponentide pakkimise tehnoloogia, OpenDoc'i osad võivad olla pakitud binaarsel kujul (DLL). Kuna SOM toetab nii liidese kui ka realisatsiooni mitmest pärimist, võib uusi osi tuletada olemasolevatest.

Bento

Bento määratleb konteineri vormingu, mida saab kasutada failides, võrgu voogudes, lõigetelaudades, jne. Bento konteineri formaat on platvormist sõltumatu ja võib salvestada suvalist tüüpi andmeid. Bento dokumendis on igal objektil püsiv identifikaator, mis identifitseerib teda erinevates süsteemides. Samuti toetab Bento erinevates dokumentides olevate objektide vahelisi viiteid. Kui dokumendist eksisteerib mitu versiooni, salvestatakse Bento's ainult versioonide vahelised erinevused. Bento't võib samuti kasutada objektide ja nendega seotud stsenaariumite nihutamiseks, mis teeb temast sobiva tehnoloogia mobiilsetele agentidele.

Uniform Data Transfer

...

Liitdokumendi Haldur (*Compound Document Management*)

Liitdokumendi haldur määratleb protokolli, mis lubab osadel jagada visuaalset pinda ja koordineerida jagatud ressursside nagu klaviatuur, menüüd ja fookus, kasutamist.

Osad on OpenDoc'is fundamentaalsed, igal dokumendil on kõige ülemisel tasemel osa, milles sisalduvad kõik teised osad.

Raamid (*frames*) on ekraani alad, mis esitavad osa. Raamid esindavad osa ka läbirääkimistes visuaalse ruumi pärast dokumendi laotuse ajal.

Osad on seotud osaredaktoritega, mida võib töö ajal valida. OpenDoc julgustab tarkvaratootjaid pakkuma koos dokumendiga vabalt jagatavaid vaatlejaid (*viewers*) mis lubavad kuvada osa aga ei luba muuta tema sisu.

OpenDoc'is on osa ja dokumendi vaheline erinevus kaotatud. Näiteks kui lohistada ikoon, mis esitab dokumenti ja kukutada ta avatud dokumendi sisse, muutub see dokumendi koopia sisalduvaks osaks. Samuti kui lohistada mingi osa dokumendist ja kukutada ta töölauale, saame eraldi dokumendi, mida esitab ikoon.

Teine tähtis visuaalne eripära on suvalise kujuga osade toetus. See tähendab, et osad ei pea olema esitatud nelinurksel ekraanipinnal -- võib luua osi, mis meenutavad oma tegelikus elus esinevaid eeskujusid.

OSA (Open Scripting Architecture)

Avatud stsenaariumite arhitektuur (OSA) on laiendus Macintosh'si *Apple Events*'ist. OSA lubab osadel paljastada oma sisu semantiliste sündmuste abil, mida suvaline OSA'le vastav stsenaariumikirjelduse keel võib välja kutsuda. Käsud, mida saadetakse nende semantiliste sündmuste kaudu tegutsevad objekti spetsifikaatori abil, mis identifitseerib objekte, mida kasutaja ekraanil näeb. Kuna on tegu objektidega, on

käsud polümorfsed.

Stsenaariumivalmis osa peab suutma töö ajal vastata päringule endas sisalduvate teiste osade kohta ja operatsioonide kohta, mida ta on võimeline teostama. OpenDoc lubab ühendada stsenaariume semantiliste sündmustega, tehes võimalikuks stsenaariumite abil muuta osa käitumist.

Lisaks võivad OpenDoc'i osad olla n.ö. "lindistatavad" (*recordable*). Sellisel juhul osa püüab kinni kõik enda suhtes toimuvad tegevused, muundab need semantilisteks sündmusteks ja saadab siis endale tagasi -- seda nimetatakse pudelikaelaks (*bottleneck*). Pudelikaela saab kasutada kõikide sündmuste "lindistamiseks". Sellist lindistust võib hiljem, kui ta on teisendatud mingisse OSA stsenaariumikeelde, hiljem "maha mängida".

OpenDoc klient/server süsteemides

OLE (*Object Linking and Embedding*) Liitdokumendi mudel

Algatuseks on OLE mittevastavalt oma nimele täielik keskkond komponentidel põhineva tarkvara jaoks, võisteldes sellega CORBA ja OpenDoc'iga. Samutikui CORBA, sisaldab OLE hulka ühiseid teenuseid, mis lubavad komponente intelligentselt koos kasutada. Ainus erinevus CORBA'st on OLE's puuduv toetus hajutatud komponentidele.

Alates OLE 2'st kasutab OLE objekttehnoloogiat nimega COM (*Component Object Model*). OLE areneb kahes suunas:

Liitdokumendid on töölaual elavate visuaalsete komponentide raamistikuks;
COM on objektsiiniks ja pakub ühiseid teenuseid.

1994 aastal lisas Microsoft uue visuaalsete osade (*custom controls*) arhitektuuri OCX. OCX on OLE versioon üldisest osast, mida saab lülitada liitdokumendi raamistikku.

OLE pole eraldiseisev toode, vaid ta on Windows operatsioonisüsteemi osa. Sealjuures kõik süsteemi teenused (ka failisüsteem) on realiseeritud COM objektidena. Sellise ülesehituse tõttu osutuvad operatsioonisüsteem ja liitdokumendi kasutajaliides laiendatavateks.

OLE koosneb hulgast liidestest (*interface*) ja Win32'le vastavatest API'dest. OLE liidesed on samalaadsed CORBA liidestele -- nad määratlevad seotud funktsioonide hulga. Liides määrab komponentide vahelise lepingu (*contract*). OLE komponent võib toetada ühte või mitut liidest. COM sisaldab liideste läbirääkimise protokolli, mis lubab kliendil töö ajal kindlaks teha, milliseid liideseid komponent toetab.

OLE komponendi määravad klass (*class*), mis realiseerib ühte või mitut liidest ja klassi tehas (*class factory*), mis on liides antud klassi isendite (*instance*) tootmiseks. Erinevalt OpenDoc'i osast pole OLE komponent ettemääratud iseeneses sisalduv üksus.

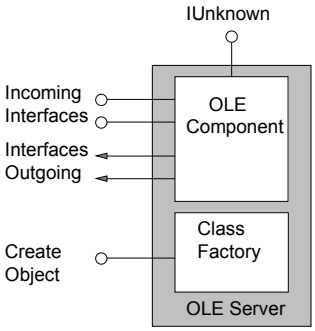
OCX osad omavad hulka ettemääratud liideseid nagu OpenDoc'i osad. Puudusena võib märkida, et OCX osa ei või OpenDoc'i osa või OLE konteineri kombel sisaldada teisi osi. OCX on hulk lepinguid OLE visuaalse komponendi ja tema konteineri vahel.

OLE koosneb:

- objektisiin COM (OpenDoc'is SOM)
- liitfailid (OpenDoc'is Bento)
- ühtne andmevahetus
- liidokumentid haldamine
- stsenaariumiarhitektuur ja automatiseerimine

COM (Component Object Model)

COM määratleb liidesed komponentobjektide vahel ühes rakenduses või rakenduste vahel. COM'i objektmudel ei toeta mitmest pärimist. Selle asemel võib COM komponent toetada mitut liidest, kasutades sisalduvust (*aggregation*). Kokkuvõttes on COM komponent "lame", "pärimine" saavutatakse viitade struktuuri abil, mis ühendavad erinevaid liideseid.



OLE automatiseerimine ja stsenaariumikirjeldus

OLE automatiseerimise ja stsenaariumikirjelduse teenused lubavad juhtida serveri komponente automatiseerimisklientide (kontrollerite) poolt. Automatiseerimiskontrollerit juhib tavaliselt stsenaariumikirjelduskeel või Visual Basic'u laadne tööriist. Automatiseerimine põhineb OLE dünaamilisel väljakutsemeetodil, mille nimi on edastavad liidesed (*dispatchable interfaces*). nagu CORBA dünaamiline meetodikutse võimaldab edastav liides kutsuda välja meetodit või manipuleerida omadusega kasutades hilist seostamist (*late binding*). Edastus ID antakse meetodile *invoke*, mis töö ajal leiab, millist meetodit kutsuda.

OLE ühtne andmevahetus

OLE strukturrne andmehoidla

OLE struktuurne mälusüsteem (*Structured Storage System*) realiseerib failisüsteemi failis. Selle realisatsioon kannab nime liitfailid (*compound files*) -- varasem *DocFiles*. Liitfailid lisavad olemasolevale failisüsteemile ühe taseme, jagades faili hulgaks salvestiteks (*storages*) -- sisukorrad, ja voogudeks (*streams*) -- andmeväärtused. Seda sisemist sisukordade süsteemi saab kasutada dokumendi sisu organiseerimiseks. Liitfailid pakuvad algelisi transaktsioonide teostamise vahendeid.

OLE sisaldab liideseid, mis lubavad kliendil suhelda püsiva (*persistent*) objektiga ja määravad püsiva objekti omadused.

OLE pakub püsivat nimeteenust, mida nimetatakse hüüdnimedeks (*monikers*). Hüüdnimi on intelligentne nimi, mida võib siduda püsivate andmetega. Kõikidel hüüdnimedel on sama liides aga neil võivad olla erinevad sidumisalgoritmid nimega seotud andmete leidmiseks. Näiteks võivad hüüdnimed olla kaugfailil, faili sees oleval elemendil, SQL päringul, mis leiab andmed relatsioonilisest andmebaasist.

OLE Liitdokumendi teenus

OLE liitdokumendi teenus määratleb liidese konteiner-rakenduse ja server-komponentide vahel. Konteineri/serveri liides määratleb serverite aktiveerimise ja nende paiga peal redigeerimise (*in-place editing*) protokollid. Konteiner-rakendus haldab mälu ja liitdokumenti kuvavat akent. Iga server-komponent juhib oma andmeid. Serveri andmed on manusandmed (*embedded*) kui nad on salvestatud konteineri liitdokumenti.

OCX'id on teatud tüüpi serverid, mis ühendavad OLE automatiseerimise ja liitdokumendi tehnoloogia. OCX'id on manussedrverid, mis toetavad paiga peal redigeerimist.

	OpenDoc (CILabs)	OLE (Microsoft)	Taligent	NeXT
Objektsiin	SOM (*Systems Object Model*) CORBA	COM (*Component Object Model*)		PDO (*Portable Distributed Objects*)
Andmehoidla	Bento	liitfailid (*compound files*)		
Ühtne andmevahetus				
Liitdokumendi haldamine				
Stsenaariumid ja automatiseerimine	OSA -- semantilised sündmused	edastavad liidesed		