**14. Управление на конфигурациите. Автоматизация - CASE средства.**

* **Управление на софтуерните конфигурации = УСК**

**(Software Configuration Management = SCM)**

Под **конфигурация** се разбира състоянието на компютърна система (или нейни компоненти), така както е определено чрез броя, естественото и взаимосвързаността на съставящите я части.

Под **софтуерна конфигурация**се разбира съвкупността от всички елементи, необходими за функционирането на даден ПП.

Това са първични текстове на програмите, обектен и изпълним код, командни файлове или процедури, необходими за свързване и изпълнение на програмната система, използвани системни файлове, помощни средства, файлове с данни и др. Изключително важна е съпровождащата документация, включваща описание на програмите, и експлоатационната документация, включваща ръководство за потребителя, ръководство за инсталиране и др.

**УСК** еглобална дейност, използаваща технически и административни похвати, за да идентифицира и документира функционалните и физически характеристики на всеки елемент на конфигурацията; да контролира промените на тези характеристики; да регистрира и съобщава достигнатия етап на внасяне на изменения, както и да проверява съответствието с определени изисквания.

Необходимо е разграничаване на съпровождането на **УСК**. Съпровождането обхваща само дейности, свързани с внасяне на промени във внедрен софтуер. Управлението на софтуерната конфигурация е фонова (umbrella) дейност през целия ЖЦ.

**УСК** се осъществява чрез следните четири дейности:

1. **Идентифициране на софтуерната конфигурация** – определяне на общата

структура на разработвания продукт, избиране на елементите на конфигурацията и регистриране на техните функционални и физически характеристикив техническата документация. Избраните елементи на конфигурацията трябва да съответстват на структурата на продукта, да представляват интерес за проследяване на развитието им и да имат уникален идентификатор (label). Препоръчва се идентификаторът да се състои от 2 части – постоянно име и номер на версията.

1. **Контролиране на софтуерната конфигурация** - оценяване, координиране,

одобряване/отхвърляне и реализиране на промени на елементите от конфигурацията след формалното им идентифициране.

За всеки ПП съставеното описание на софтуерната му конфигурация задава и връзките между елементите й, така че при всяка промяна в един елемент да може да се проследи кои други елементи са засегнати. Осъщесвява се управление на заявките за изменение, контрол на версиите и управление на внасянето на измененията.

Обикновено се поддържа стандартен формуляр — заявка за изменение, в който се описва исканото изменение и се обосновават причините за него. Специален експертен съвет разглежда постъпилите заявки и за всяка от тях определя дали да се отхвърли, дали да се включи в подготвяната нова версия или да се отложи за някаква следваща версия. **Критерии за групиране измененията на поредна версия могат да бъдат:**

**а)** подреждане на измененията по технически или процедурни причини;

**б)** подреждане по неотложност на извършване на промените;

**в)** оценяване на необходимите ресурси за реализиране на измененията;

**г)** анализиране степента на отражение на промените върху останалите части  
от системата. Би трябвало да се минимизира броят на засегнатите от измененията програмни части, като тези, от които зависи работоспособността на системата, да се променят само след доказана необходимост и наличие на ресурси;

**д)** сходните промени да се групират в една версия.

Задължително е реализирането на приемственост между версиите. Това означава, че всяка нова версия трябва да реализира всички функциите на предишната и евентуално да предлага нови.

Трябва да се осигури и пълно съответствие между ПП и документацията му. Обикновено се поддържа бюлетин за измененията. В него на достъпен за потребителите език се описват промените в новата версия. Ако броят на бюлетините е много голям, се препоръчва преиздаване на потребителската документация.

Чрез контролиране на софтурната конфигурация се предотвратяват:

- многократното съпровождане при всеки поребител (откритите дефекти трябва да се отстраняват във всички копия, и то по един и същ начин);

- едновременно обновяване на един и същ елемент от различни разработчици;

- неправомерно променяне на общи елементи като подпрограми, класове, заглавни файлове, библиотеки и др.;

- неправилно управление на версиите.

**3) Следене на състоянието на софтуерната конфигурация** (Status Accounting) – регистриране на заявките и извършените промени в протоколи (logs) и архиви и отразяването им в съответни справки (за транзакциите, за променитем за развитието на елемент от конфигурацията, за използваните ресурси и т.н.)

**4) Проверяване на софтуерната конфигурация** (Configuration Auditing) – за спазване на регламентираните процедури и за качеството на създаваните версии – междинни (baselines) и поребителски (releases). Проверяваните изисквания съответстват на броя, размера и типа на разработваните елементи на софтуерна конфигурация. Проверките се реализират чрез наблюдения, въпросници и разговори. Те се провеждат регулярно в определени контролни точки на проекти, обикновено от квалифицирани специалисти, неучастващи в разработката.

* **Автоматизация - CASE средства.**

**АСП** = **Aвтоматизация на софтуерното производство**

Предназначението на автоматизиращите средства е да поддържат избраните методи за разработване на софтуер, да улесняват управлението на проектите и на цялостния процес на създаване на ПП, да извършват трансформации от едно представяне на софтуерните продукти в друго и да проверяват правилността им. В зависимост от прилаганите автоматизиращи средства могат да се разграничат два основни подхода —използване на индивидуални средства и използване на интегрирани среди. Ще се спрем на основните характеристики и особености на всеки от тези два подхода.

**Абревиатурата** **CASE** (**Computer-Aided Software Engineering** - автоматизирана разработка на **ПО**) означава специален тип програмно осигуряване, предназначено да подпомага разработката на софтуера и процесите на развитие. Затова към **CASE средства** се отнасят редакторите на проектите, речници, компилатори, средства за построяване на програмни системи и т.н. **CASE** технологиите предлагат поддръжката на процеса на разработването на **ПО** по пътя на автоматизация на някои етапи на разработка, а така също на създаване и предоставяне на информация, необходима за разработването. Процеси, които могат да се автоматизират с помощта на **CASE** **средствата**:

**1.** Разработка на графични модели на системата на етапите на създаване на спецификацията и проектирането.

**2.** Проектиране на структурата на ПО с използване на речникови данни, съхраняващи информация за обектите на структурата и на връзките между тях.

**3.** Генериране в диалогов режим на потребителски интерфейс на основа на неговото графично описание.

**4.** Настройване на програмата на базата на информацията, която се получава в хода на изпълнение на програмата.

**5.** Автоматична транслация на програми, написани на остарели езици за програмиране(например COBOL), в програми, написани на съвременни езици. В настоящия момент за болшинството процеси, които се изпълняват в хода на разработката на ПО, съществуват подходящи CASE технологии. Това води към определено подобряване на качествата на създаваните програми и повишаване производителността на труда на хората, разработващи програмното осигуряване. Заедно с това тези постижения отстъпват на очакванията, които са присъствали при зараждането на CASE технологиите. Тогава се е смятало, че щом се внедряват CASE средствата, може да се постигне значително повишаване и на качествата на програмата, и на производителността на труда. Фактически това повишение е примерно 40%. Това повишение се смята за значително. CASE технологиите не извър

**CASE класификация:** Класификацията помага за разбирането на различните типове CASE средства и как те подпомагат дейностите в процеса.

* Функционална перспектива - класификация според специфичните функции
* Перспектива от гледна точка на процеса - класификация според

подпомаганите дейности

* Интеграционна перспектива - средствата се класифицират според това как

са организирани в интегрираните единици, които подпомагат една или повече дейности.

**Автоматизация чрез индивидуални средства**

Този подход е исторически първият и се състои в използването на отделни (stand-alone) "полезни" програми, улесняващи извършването на някаква дейност при създаването на софтуера. Първоначално средствата са били компилатори, асемблери, свързващи редактори, дебъгери, осигуряващи директна помощ за програмирането.

Съществуващите индивидуални средства за **АСП** могат да бъдат класифицирани по различни критерии — по функциите им (т. е. кои дейности подпомагат), по използването им в различни фази от жизнения цикъл, по основните им потребители (мениджъри, програмисти, отговорници по осигуряване на качеството и др.), по степента им на сложност на усвояване и прилагане и др. В зависимост от автоматизираните дейности те се разделят на следните групи:

**а)** средства за програмиране — езиковоориентирани редактори, транслатори, свързващи редактори, дебъгери, средства за управляемо изпълнение на програми и др.

**б)** средства за тестване — анализатори на програми, генератори на тестови данни,

средства за управление на тестването

**в)** средства, подпомагащи управлението на проекти. Чрез тези средства мениджърът на софтуерната разработка може:

— да оценява предварително трудоемкостта, стойността и продължителността на

проекта и броя на хората, необходими за реализацията му. Оценката се прави чрез въвеждане на индиректна мярка за размера на проекта и анализиране на някои общи

характеристики — сложност на проблема, опит на разработчиците в тази приложна

област, зрелост на процеса на разработване и др.;

* да определя основните задачи и да създава графици за изпълнението им;
* да проследява развитието на проекта и при необходимост да го препла-

нира с промяна на ресурсите и сроковете;

* да оценява производителността на труда и качеството на създавания

продукт чрез прилагане на подходящи метрики;

— да проследява удовлетворяването на изискванията,  
**г)** средства, подпомагащи документирането

Те осъществяват създаването, оформянето и отпечатването на документи. В тази група са текстообработващите програми с възможности за въвеждане, редактиране, проверка на граматическата правилност и стил на текст. Графичните редактори позволяват илюстрирането на текста с диаграми, схеми и произволни изображения. Издателските системи реализират форматирането (предпечатната подготовка) на оформените в съответствие с определени стандарти документи.

**д)** средства, подпомагащи съпровождането:

* за управление на софтуерните конфигурации — идентифициране, контрол на версиите и управление на внасянето на изменения;
* статични и динамични средства за възстановяване на детайлния проект по изходните текстове на софтуерната система с цел повторно разработване или промяна, за да се подобрят някои характеристики на софтуерната система (reverse engineering и re-engineering);

**е)** средства за анализ и проектиране

Те позволяват създаването и оценяването на модел на софтуерната система, която ще се разработва. Могат да поддържат различни методи на проектиране — структурни или обектно ориентирани.

**ж)** средства за прототипиране и симулиране

Тези средства представят някои функции или характеристики на поведението на софтуерни системи, работещи в реално време.

**з)** средства за проектиране и разработване на потребителския интерфейс;

**и)** езикови процесори.

Тези средства са за използване на различни езици — за специфициране, за описание на проекти, за автоматично генериране на текста на програмите и т. н.

Изследвания за използване на индивидуални средства за **АСП** показват, че разпределението им по различните фази и функции от жизнения цикъл не е равномерно и че успехът на нови методи в софтуерното производство зависи в голяма степен от това, дали тези методи се подпомагат и от съответни автоматизиращи средства. Основно предимство на подхода за **АСП** с индивидуални средства е, че те не са толкова скъпи и всяка софтуерна фирма може да си ги позволи. Освен това тези средства улесняват една или няколко дейности и повишават производителността на персонала, участващ в създаването на софтуер. Недостатъците на подхода са няколко. Преди всичко индивидуалните средства са обикновено хардуерно зависими и са предназначени за точно определена операционна и програмна среда. Използването на няколко независими средства принуждава разработчиците да разучат няколко различни потребителски интерфейса. Липсата на интеграция намалява производителността, защото не е възможно последователно извикване на някои средства, без да има дублиране на общите дейности.

**Автоматизация чрез интегрирани среди**

Полезността на индивидуалните средства за **АСП** е безспорна, но **интегрирането** им за съвместно използване би улеснило:

* предаването на информация между тях;
* ефективното изпълнение на глобални дейности, като документиране, осигуряване на качеството и управление на софтуерните конфигурации;

— реализацията на потребителски сценарии за решаване на конкретен проблем.

Формулирани са следните **изисквания към интегрираните среди за** **АСП**:

* да осигуряват механизъм за общо използване на информацията от всички средства в средата;
* да допускат директно извикване на всяко средство;
* да поддържат решаването на всяка конкретна задача при реализацията на софтуерния проект чрез подходящо съчетаване на средства;
* да улесняват комуникациите между всички участници в процеса на създаване на софтуер;
* да натрупват статистическа информация, която да се използва за подобряване качеството на процеса и продукта.

**В зависимост от начина на свързване на група от автоматизиращи средства в единна среда интеграцията може да бъде:**

**а)** еклектична интеграция — съществуващи индивидуални средства се обединяват в система чрез създаване на програма—монитор, извикваща всяко от средствата;

**б)** интеграция чрез данните — използване на общ модел на данните. Този вид интеграция може да бъде с различно ниво на сложност — обмен на данни между две средства чрез програма — конвертор, използване на обща съвкупност от прости символни файлове или чрез система за управление на обекти;

**в)** интеграция чрез потребителския интерфейс. В този случай средствата в системата използват общ стил и съвкупност от общи стандарти за връзка с потребителя;

**г)** интеграция чрез дейностите, които се поддържат.

Този тип интеграция се основава на модел на процеса, който определя основните извършвани дейности, резултатите от тях, потока на данни и потока на управление. Известно е и кои средства в интегрираната среда кои основни дейности поддържат.

**Видове интегрирани среди**

Всяка интегрирана среда обединява в едно положение няколко софтуерни средства, поддържащи специфични дейности в процеса на създаване на софтуер. Чрез интеграцията се постига:

* хомогенен и логически последователен потребителски интерфейс;
* лесно извикване на всяко средство и на верига от средства;
* достъп до общо множество от данни, поддържани по централизиран начин.

Удобствата за работа, предоставяни от някои среди за АСП, могат да допринесат за

въвеждане в софтуерната организация на нови методи и техники.

**В зависимост от предназначението си интегрираните среди могат да бъдат за:**

**а)** планиране и моделиране на бизнесинформационни системи. Този клас включва

продукти, подпомагащи идентифицирането и описанието на сложни бизнесдейности.

Те се използват за построяване на обобщени модели на предприятие, за да се оценят

общите изисквания и информационни потоци и да се определят приоритетите в

разработването на информационните системи. Средствата, интегрирани в такива

продукти, включват графични редактори (за създаване на диаграми и структурни

схеми), генератори на отчети и генератори на справки за срещането на отделни

елементи.

**б)** анализ и проектиране.

Съвременните средства автоматизират най-често използваните подходи за анализ и

проектиране — структурния, обектно ориентирания и подхода на Джаксън. Те

обикновено включват един или повече редактора за създаване и модифициране на

спецификации и други средства за анализирането, симулирането и трансформирането

им.Възможностите на средствата от този клас зависят от:

* нивото на формализираност на използваната нотация. Ако тя е неформална

(структуриран английски или друго свободно текстово описание), то се осигурява само редактиране и съставяне на документ. Ако нотацията е полуформална (без точна семантика, но да е възможно да се проверява синтаксисът) или с формално определени синтаксис и семантика (крайни автомати или мрежи на Петри);

* от вида на приложението — дали преобладава обработката на данните, както е в

банковите и счетоводни системи, или управленските функции;

**в)** създаване на потребителски интерфейс

Смята се, че потребителският интерфейс е определящ за пазарната реализация и използване на всяка софтуерна система. Затова са създадени интегрирани среди, които позволяват на разработчика да създава и да тества лесно компонентите на потребителския интерфейс и да ги свърже с приложната програма. Те включват:

* графични редактори за създаване на прозорци, диалогови кутии, икони и др.**;**
* симулатори за тестване на създадените компоненти преди интегрирането им с приложението;
* генератори на първичен текст;

— библиотеки за поддържане на генерирането на изпълним код.  
**r)** програмиране

Те включват текстов редактор за създаване и променяне на текста на програмите, компилатор, свързващ редактор и дебъгер. Характеризират се с:

— удобен потребителски интерфейс;

— управление на създаваната по време на сесия информация — файлове с първичен текст, междинни, обектни и изпълними файлове. За ускоряване на съставянето и тестването на програмата се съчетават компилатор с интерпретатор и свързване с отчитане на направените промени;

**д)** верификация и валидация

Такива среди подпомагат модулното и системно тестване и обикновено включват:

* статичен анализатор за създаване на управляващ граф на програма и граф на извикванията;
* средство за инструментиране на програмата и за проследяване (трасиране на изпълнението) при динамичен анализ;
* генератор на тестови данни;
* управляваща програма — средство за реализация на тестването, което създава, съхранява и поддържа тестови данни, сценарии, резултати и документация.

**е)** съпровождане

Средата за съпровождане управлява внасянето на промени, създаването и контрола на версии и за управление на софтуерните конфигурации.

**ж)** управление на проекти

Подпомагат се планирането, съставянето на графици и текущото следене на изпълнението на проекти.

**Принципи на изграждане и архитектура на интегрираните среди**

Предназначението на **АСП**-средата е да подпомага цялостния процес на създаване на ПП, като поддържа *хранилище* (repository) на информацията, необходима за осъществяване на софтуерните разработки.

Поддържането и използването на информацията от хранилището се осъществява чрез **интегрираща архитектура***.* Основните компоненти са: база от данни, в която да се съхранява информацията, система за управление на обекти, чрез която да се управлява променянето на информацията, механизъм за управление на инструменталните средства (за координиране на използването им) и потребителски интерфейс. Повечето модели представят тези компоненти като слоеве.

**Потребителският интерфейс**осигурява удобна и ефективна работа със системата.

**Протоколът на представяне**е множество от указания, чрез следването на които всички АСП-средства в системата се използват по подобен начин. Така екраните имат едно и също разпределение на отделни области с еднакво предназначение; има правила за имената и организация на менюта, икони и обекти; стандартизирано е използването на клавиатура и мишка; описан е общ механизъм за достъп до средствата. С използването на протокол на представянето се постига унифицираност.

**Слоят на индивидуалните средства**включва самите средства и програми за

управлението им. Препоръчва се управляващите програми да са проектирани и реализирани така, че да се поддържа динамична съвкупност от индивидуални средства.

Това би позволило на потребителя да съставя нужната му конфигурация от средства и да я променя, като добавя, изключва или модифицира някои средства. Ако в системата се поддържа многозадачна работа, управляващите програми осигуряват синхронизация при прилагане на средствата, регулиране на потока на данните, следене на правата за достъп и прилагане на метрики за ефективността на използване на всяко от средствата.

**Слоят за управление на обекти**осъществява интегрирането на средствата с

данните от хранилището и управлението на софтуерните конфигурации. Той е съвкупност от програми, които за всяка заявка идентифицират, обектите от съответната софтуерна конфигурация и ги представят във вид, подходящ засъответното средство; поддържат различните версии, управляват планирането, внасянето и документирането на промени, регистрация и поддържане на описание на всеки елемент на софтуерна конфигурация.

**Слоят за управление на хранилището**включва базата данни на системата и

функциите за управлението й. Идеята за централизирано съхраняване на данните присъства във всеки модел на интегрирана среда, макар и под различни имена: **АСП**-база

от данни, база на софтуерните разработки, хранилище и др. Съхраняваната информация

може да се раздели на две: обща информация (свързана със софтуерната фирма като

цяло и стила на работа в нея) и информация за всеки конкретен софтуерен проект.

Общата информация може да включва описание на организационната структура в

софтуерната фирма, съвкупността от вътрешни правила и стандарти, които трябва да се

спазват, описание на използваната методология на разработване, процедури за

извършване на основни дейности и др.

**Информацията за конкретен проект може да бъде:**

* елементи на софтуерните конфигурации (първичен текст на програми, обектни модули, описания за свързване, изпълними програми, описание на връзките между елементите на дадена софтуерна конфигурация и др.);
* информация за провежданите дейности по осигуряване на качеството (планове, сценарии и резултати от тестване и проверки; резултати от прилагане на метрики, резултати от статистическа обработка на данните);
* информация, свързана с управление на проекта — планове, оценки на трудоемкост и продължителност, графици, отчети от проследяване на проекта в определени контролни точки);
* документация — съпровождаща и потребителска.

Съдържанието и начинът на организация на данните в хранилището се определят при конкретната реализация на концептуалния модел.

Освен обичайните функции на СУБД към компонентата за управление на хранилището са формулирани някои допълнителни изисквания:

* да осигурява интегрираност на данните — да проверява всички елементи така, че да не се допуска дублиране (редундантност) на данните; да осигурява съвместимост между свързаните обекти, автоматично да извършва последователните модификации, когато промяната на един обект изисква промяна на всички свързани с него обекти;
* да осигурява механизъм за използване на информацията от множество разработчици и различни инструментални средства; да управлява многопотребителския достъп до данните и чрез защитни механизми да предотвратява наслагване на промените;
* да осигурява интеграция на данните и индивидуалните средства в системата чрез поддържане на механизми за трансформиране, управляем достъп и защита.

Така се постига съхраняването и обработването на сложни структури от данни, които се използват ефективно от съответните инструментални средства в системата, и то така, че разработването на ПП става в съответствие с избраната методология.

**Практическото използване на софтуерните среди преминава през следните**

**шест фази:**

**а) избор на софтуерна среда**

През тази фаза трябва да се направи проучване на софтуерния пазар и да се избере софтуерна среда, която е най-подходяща за дадена организация. Основните критерии за избора са:

* методологията на разработване на софтуер в конкретната организация, т. е. съвкупността от прилагани модели, методи, техники и стандарти. В някои случаи чрез използването на определена среда може да се премине от стихийно разработване към систематично и организирано разработване с придържане към поддържаната от софтуерната среда методология;
* наличния хардуер в софтуерната организация. Обикновено използването на софтуерните среди изисква значителни изчислителни ресурси, които да не могат да бъдат осигурени от компютрите, с които разполага организацията;
* приложната област, в която ще бъде създаваният софтуер;
* цената на средата. По данни от [1] въпреки обещаваното увеличение на производителността на труда от 40—100% цената за закупуване, инсталиране, настройване и усвояване на работата със средата може да бъде неприемливо висока.

**б) настройване на средата към специфични за организацията изисквания**

През тази фаза трябва да се създаде версия на средата, която да съответства на използваната в организацията платформа и модел на процеса на разработване. Това изисква определяне на стойностите на параметри, определяне на елементите на системата за управление на обекти, избор на съвкупност от средства и цялостно документиране на получения вариант на средата.

**в) инсталиране и експериментално използване на средата**

Организира се обучение и пробно използване, за да се оцени полезността на средата. Обикновено трябва да се преодолява съпротивата на разработчиците заради променения стил на работа и съпротивата на мениджърите заради големите инвестиции с неизвестна възвращаемост.

**г) използване и еволюция на средата**

Препоръчва се натрупване на статистически данни, въз основа на които да се оцени ефектът от използване на средата. Поддържането на обратна връзка с доставчиците би осигурило подобряване на функционирането й и своевременно получаване на нови версии.

**д) преустановяване на използването на средата**

Възможни са две ситуации — отказ от използване на автоматизиращи среди или замяна на използваната среда с друга. Във втория случай заради съпровождането на разработени вече системи може да се наложи паралелно функциониране на двете среди.

**Преимущества и недостатъци на автоматизацията чрез интегрирани среди**

**Потенциалните ползи от прилагането на средства за автоматизация са:**

* повишаване на систематичността и управляемостта на софтуерните проекти;
* намаляване на стойността на разработването и особено на стойността

на съпровождането;

* подобряване на качеството на софтуера;
* ускоряване на процеса на разработване, т. е. намаляване продължител-

ността на софтуерните проекти;

* повишаване на производителността на труда;
* препоръчвани техники стават реално използваеми поради подпомагането им

от софтуерни средства.

**Независимо от тези преимущества АСП-средите имат все още ограничено**

**използване. Основни причини за това са:**

* няма публикувани изследвания за резултатите от практическото използване

на средите;

* няма натрупани данни и подходящи метрики за измерване, как средите

влияят на производителността на софтуерните разработчици;

* средите са големи и сложни софтуерни системи. Те изискват значителни

инвестиции за закупуване на самите среди, за подготовка и осъществяване на внедряването им. Функционирането им е свързано с големи изчислителни ресурси. Всички тези изисквания не могат да бъдат удовлетворени в неголемите софтуерни организации.

Сложността на съвременните софтуерни системи изисква автоматизация през различните фази на разработването. Независимо от трудностите при приспособяване на автоматизиращите средства към стила на работа в конкретната организация бъдещето на софтуерната индустрия е немислимо без създаването и усъвършенстването на **АСП**-средствата.