МОДЕЛИРАНЕ НА ПРОЦЕСИ

1. Моделиране на процеси.

Моделът на процес представлява абстрактно описание на начина на разработване на софтуера (информационната система), представено от определена гледна точка. В зависимост от гледната моделите на процесите се разделят на:

- **Модел на потока от данни (dataflow model)** – описва дейностите, които са свързани с преобразуване на данни.
- **Модел на потока от дейностти (workflow model)** – описва последователността от дейности, които се извършват в процеса на разработка на системите. Всяка дейност се описва с входни и изходни параметри и връзки (зависимости) с други дейности. Най-често тези дейности се извършват от хора.
- **Модел на роля/действие (role/action model)** – описва отделните роли на участниците в разработката на системите, както и дейностите, които всяка роля трябва да извършва.

Моделите на процесите могат да бъдат **описателни** и **предписателни**. **Описателните модели** описват историята на разработването на програмните системи. Те са специфични за конкретните системи. **Предписателни модели** показват как трябва да се разработи нова програмна система. Те се използват като рамки за организиране и структуриране на дейностите и тяхното изпълнение във времето, свързани с проектиране и разработка на софтуерните системи.
Едни от известните и използвани предписателни модели са:

Каскаден модел (модел на водопада)

Този модел предлага последователен подход при разработването на софтуерните системи. При него разработването на софтуер е разделено на пет основни стъпки/ дейности:

1. събиране на изискванията към системата;
2. планиране – изготвяне на график;
3. анализ и проектиране;
4. реализация (генериране на код) и тестване;ю
5. внедряване и поддържане.

Моделът на водопада има следните характеристики:
-всяка дейност трябва да бъде завършена, преди да се премине към следващата;
-всяка стъпка завършва с документиране (създаване на необходимите документи);
-ясно определени роли на разработчиците на системата;
-ясно разграничен процес на разработване, лесен за разбиране;
-не е гъвкав (трудно се адаптира към променящи се изисквания на клиента в процеса на разработката).

Модел на бързата разрабокта (RAD-Rapid Application Development)

Този модел има за цел да се намали времето за разработка на софтуерната система. Базира се на модела на водопада и възможността дейности 3 и 4 (виж каскаден модел) да се реализират паралелно от екипа разработчици. Например, всеки от епипа или един екип да проектира и разработва модул и/или различна фунцкионалност от крайния продукт. Намаляване навремето за разработка може да се постигне чрез средства за автоматично генериране на код, както и чрез имплементиране на готови софтуерни компоненти.

Еволюционни (фазови) модели

Фазовите модели могат да бъдат постъпкови (инкрементални) и итеративни.

Инкрементален (постъпков) модел

При инкременталният модел разработката на системата се реализира на стъпки, като всяка стъпка се доставя само част от цялата функционалност. При него на клиента се предоставят последователност от версии, наречени инкременти. Всяка следваща версия има повече функционалностти от предходната.

Итеративен модел

При този модел на клиента се доставя цялостно решение (система). На всяка следваща итерация се доставя все по-добра (по-завършена и с по-добро качество) версия на софтуера.

Фазовите модели имат следните характеристики:

- на всяка стъпка се доставя част от функционалността на системата и клиентът може да я използва, преди да е готова цялата система;
- първите стъпки могат да служат като прототип, чрез които да се определят изискванията към следващите стъпки от реализацията на системата;
- по-малък риск от неуспех на целия проект;
- позволява използването на нови технологии в самия процес на разработка.

Прототипен модел

Основната цел на прототипния модел е създаване на прототип, които да извлече или валидира изискванията към системата. Има два типа прототипи – еволюционен и хвърлен (throwaway). Целта на еволюционния прототип е да достави на потребителя работеща система, а на хвърления - да подпомогне специфицирането на изискванията към софтуера.

Прототипният модел може да се използва самостоятелно и в комбинация с другите модели. Неговото използване е удачно в проекти, за които не са достатъчно ясни потребирелските изисквания и дизайна на системата.

Спираловиден модел

Този модел се представя чрез спирала. Всяко пълно завъртане по спиралата представлява една стъпка от разработката. Специфична негова характеристика е анализ на риска. Спираловидният модел е итеративен подход и има множество точки от прогреса (anchor point milestones). При него при всяко завъртане по спиралата се преминава през следните четири сектора:

1. Определяне на целите – определят се целите на текущата фаза от разработката;
2. Оценка на риска – идентифицират се и анализират се потенциалните рискове. Предприемат се действия за намаляването или елиминирането им.;
3. Разработка и валидиране – избира се модел за разработка на текущата фаза;
4. Планиране – анализира се текущото състояние и се планира следващото завъртане по спиралата.

Спираловидният модел е подходящ при разработване на големи (large-scale) софтуерни системи. Той може да се прилага през целия жизнен цикъл на системите.

2. Езици за моделиране на процеси.

Езиците за моделиране на процеси се използват за представяне на характеристиките на софтуерните процеси, а именно дейностите, които трябва да се извършат, ролите на участниците в процеса и средствата, които се използват. Те имат за цел подпомагане на разбирането, дизайна, симулирането, опимизирането и поддържането на процеса.

Език за моделиране на процес (Process modeling language – PML) е език създаден или адаптиран за представяне на процеси.

Класификацията на езиците за моделиране на процеси може да се направи по различни критерии/признаци.
Спрямо тяхната организация езиците за моделиране на процеси се разделят на:

- **Entity – Relation** – организирани са като същности, с връзки между тях. Използват се за моделиране на информационни системи.
- **Role – Interaction** – описват ролите и техните взаимоотношемия.
- **Object - Oriented** – основна единица при тях е обект. За всеки обект се описват данни и функционалност в един пакет.

Спрямо тяхната формалност езиците за моделиране на процеси се разделят на:

- **Формални** – представят се чрез формален систаксис и семантика;
- **Полуформални** – обикновено имат графично представяне с формален синтаксис и неформална семантика (могат да бъдат интерпретирани по различен начин);
- **Неформални** – нямат строго дефинирани правила, а смисълът на конструкциите им е от реалния опит и употреба (например говоримите езици).

За моделиране на процеси се изпозват три типа (основни парадигми) езици:

- **Дескриптивни** (логически) – използват правила или тригери.
- **Мрежово базирани** (езици на базата на мрежи) – представят процесите като мрежи, мрежи на Петри, обобщени мрежи, системи за масово обслужване и др.
- **Императивни** – базирани на езиците за програмиране. Процесът е представен като програма.

3. Шаблони за описание на процес.

Шаблон е описание на общо решение на общ проблем или въпрос, на базата на което може да се извлече детайлно решение на специфичен проблем.
Шаблон на процес е шаблон, които описва доказан успешен подход и/или последователност от действия за разработване на софтуер.
Шаблонът на процес представлява структурирано описание на процес. Той описва какво трябва да се направи, а не как трябва да се направи. Шаблоните могат да бъдат дефинирани на различни нива на абстракция. По нарастваща степен на абстрактност типовете шаблони са:
Шаблон на задача на процес (Task process patterns) – описва отделни стъпки за извършване на действие или задача, които са част от процеса;
Шаблон за етап на процес (Stage process patterns) – преставя стъпките, които се извършват в рамките на една базова дейност на процеса;
Шаблон за фаза на процес (Phase process patterns) – дефинира последователността и взаимодействието между базови дейности в рамките на процеса.
Всеки шаблон на процес трябва да съдържа следните елементи:

- ✓ Начален контекст – описват се условията, които трябва да са налице, за да е приложим шаблонът;
- ✓ Проблем – описание на проблема, които трябва да се реши;
- ✓ Решение – описва се какво прави шаблонът с цел да разреши проблема. Тук могат да се използват както текстове, така и диаграми;
- ✓ Краен контекст – какви дейностти трябва да са изпълнени, какво трябва да е състоянието на процеса, каква информация е обработена/събрана;
- ✓ Шаблони с които е свързан (има връзка) – списък на всички шаблони на процеси, които са директно свързани с този шаблон. Препоръчително е списъкът да онагледява йерархията между шаблоните;
- ✓ Начин на употреба – описание на случаите, в които шаблонът е приложим.

Шаблоните на процеси предоставят ефективен механизъм за описание на всеки софтуерен процес. Те позволяват да се разработи йерархично описание на процесите на

разработка на софтуерните системи от най-високото до най-ниското ниво на абстракция.

4. Функции и функционални дървета.

Функциите обикновено описват определени дейности или задачи в дадена проблемна област. Те доставят определени изходни данни на основата на входни данни и предизвикват промяната на съдържанието или структурата на определени данни. Като функции могат да се представят различни задачи, които трябва да изпълнява разработваната система. Функции могат да бъдат подпрограми (функции, процедури, методи) на език за програмиране, избран за реализация на системата.

Декомпозирането на задачите на по-прости функционални идентичности може да се представи като функционално дърво. То представя определена йерархия от функции (Функции→ Подфункции→...). Възможни са различни интерпретации на една такава йерархия.

Основните характеристики на функционалните дървета са следните:

- ✓ Представят концепция за систематична разработка и декомпозиция на сложни проблеми (системи);
- ✓ Дават първоначална насока за представяне на диалога между участниците в процеса;
- ✓ Представят само функционалния аспект на системите.

5. Диаграми на потока от данни.

Диаграмите на потока от данни (Data Flow Diagrams – DFDs) представят разработваната система като един информационен поток от данни между функции, памет и интерфейси. При движението данните могат да бъдат трансформирани от един в друг вид.

Съществуват различни възможности за представяне на потоците от данни. Една от тях е за базовите елементи на диаграмите да се използват следните графични символи:

- ✓ Поток от данни – представя се като именувана стрелка;
- ✓ Функция (съответства на процес) - представя се като именувана обръжност;
- ✓ Памет (място за съхранение на данни) - представя се като две паралелни линии, между които се задава името;
- ✓ Интерфейс (към външни обекти) - представя се като правоъгълник, именуван с името на интерфейса.

Базовите елементи могат да се комбинират за представяне на по-сложни структури на информационни потоци.

Приложението на диаграмите на потока от данни е свързано със следните правила:

- Всяка диаграма трябва да съдържа поне енин външен обект;
- Всеки външен обект се представя само веднъж
- Всеки поток от данни има име. Изключение правят само потоци които водят/идват към/от хранилището. Приема се, че те съдържат всички данни от него.
- Между интерфейсите към външни обекти не се представят потоци от данни;
- Между външните обекти и паметите трябва винаги да се дават функции.

Диаграмите описват потоци от данни, а не контролни потоци. Затова те не съдържат възможности за разклонения и цикли. Интерфейсите трябва да се избират така, че да дават ясна представа за оригиналния източник или предназначение на информацията. Изборът на интерфейси трябва да се обстархира от конкретния начин на въвеждане на информацията. Целесъобразно е имената на финкциите да бъдат глаголи, следвани от имената на конкретни обекти (Управление на данни за студенти, Заемане на книги и др.).

Диаграмите на потоци от данни лесно се създават и са лесни за разбиране и от непрофесионалисти. Те съдържат повече информация от функционалните дървета.

6. Диаграми на сценарии.

Сценариите се представят като диаграми, които описват:

- ✓ Биснес процесите в една информационна система;
- ✓ Връзките между бизнес процесите;
- ✓ Връзките между бизнес процесите и актьорите на системата.

Сценариите описват функционални изисквания към разработвано приложение.
Бизнес процесите обикновено се описват като сценарии за използване (use cases).
При създаване на диаграми за използване анализаторите първо идентифицират актьорите на системата, а след това и самите сценарии. Актьорите на системата са субектите на разработваната система. Те могат да бъдат един или група потребители, както и различни видове външни устройства или софтуер.
Актьорите представят определени роли, които потребителите (или устройствата) изпълняват при работа със системата. Формално те се дефинират като субекти, които комуникират със системата и са външни за нея.
Принципната разлика между потребител и актьор се състои в това, че потребителят може да изпълнява различни роли, докато на актьорът се присвоява само една.
За представяне на сценариите се изполват различни техники (инструменти), като:

- ✓ Текстово представяне (текстова схема) – шаблон на естествен език, по който трябва да се опише всеки сценарий;
- ✓ Диаграми на сътрудничество (collaboration diagrams) – представят сценариите като обекти на системата и връзки между тях. Представят се и съобщенията, обменяни между обектите;
- ✓ Диаграми на последователности (sequence diagrams) – при тях обменяните съобщения са подредени последователно във времето;
- ✓ Диаграми на действия (activity diagrams) – представят сценария като алгоритъм;
- ✓ Крайни автомати (state-transition diagrams) – представят поведението на даден обект в системата като възможни състояния, които той може да приема по време на изпълнение на сценария;
- ✓ Мрежи на Петри.