## Vývoj software a webových aplikací

Vývoj software není pouze samotná implementace, ale i příprava a následné nasazení či údržba. Je to celý proces, který se dělí do 6 základních fází.

Vztahy mezi těmito fázemi se určují pomocí tzv. modelů. Jednimi z nejznámějších jsou modely Waterfall, Spiral a Prototype. [4] Model Waterfall je specifický tím, že jednotlivé fáze „komunikují“ pouze mezi sebou. U Spiral se jednotlivé fáze cyklicky opakují a model Prototype je kombinací modelů Waterfall a Spiral, u níž dochází k opakovaným změnám prototypu ještě před zahájením fáze implementace.

### Identifikace a specifikace požadavků

Požadavky na software popisují funkcionality, které musí systém poskytovat uživatelům. Tato počáteční fáze je hodně důležitá, protože pokud jsou požadavky klienta správně identifikovány a specifikovány, dochází v pozdějším průběhu celého vývoje k minifikaci nedorozumění mezi oběma stranami a ušetří se čas i peníze na korekturách prototypů a návrhů.

Požadavky identifikujeme především dotazováním budoucích uživatelů nebo analýzou stávajícího systému či průzkumem, jaké systémy s podobnými funkcionalitami trh nabízí. Otázky uživatelům by měly zahrnovat například následující:

* Jaká je vaše role v současném systému?
* Jak současný systém používáte?
* Jaké vylepšení by jste od nového systému očekávali?

Analyzování existujícího systému zahrnuje pochopení a zdokumentování současného systému, určení, které části systému by měly být automatizované a které manuální, a dotazování uživatelů, jaké funkce by měly být řešené jinak a které přidané. [1]

Výstupem této fáze je tzv. katalog požadavků, kde každý požadavek má své identifikační číslo, popis a prioritu, případně další náležitosti.

### Analýza

V této fázi dochází k analýze požadavků, na základě které se vytvoří modely a diagramy znázorňující jednotlivé funkce systému, ať již stávající (AS-IS) nebo ty budoucí (TO-BE). Tyto diagramy slouží především pro přehlednost informací, jejich znovupoužitelnost a sledování realizace požadavků. Pro vytváření modelů a diagramů existuje unifikovaný vizuální modelovací jazyk, tzv. UML. Mezi tyto diagramy patří napřiklad diagram aktivit, tříd, případů užití, komponent, nasazení nebo stavový diagram. [6]

### Návrh

#### Architektura

Architektura jako taková obecně představuje skládání struktur. U architektury software se tyto struktury nazývají komponenty, což jsou jakési „balíčky“ obsahující související funkce či data. V rámci architektury určujeme vztahy mezi těmito komponentami a zajišťujeme, aby byla oddělena celková struktura komponent a jejich rozhraní od detailů jednotlivých komponent. Mezi příklady software architektury se řadí například SoC (Separation of Concerns), Klient-Server, Pipes and filters nebo návrhové vzory.

Software architektura společně s hardware a procesy tvoří architekturu informačních systémů. Informační systém zajišťuje organizaci získávání, zpracování a distribuce údajů k uživatelům systému.

Na nejvyšší úrovni se nachází enterprise architektura, jejíž součástí je výše zmíněná architektura informačních systémů a software architektura. Je využívána především v rozsáhlejších organizacích s větším množstvím propojených systémů. [7] Jedná se hlavně o architekturu vícevrstvou, typicky třívrstvou, což je klient-server architektura, kde je však oddělena prezentace dat, aplikační logika a datová struktura do jednotlivých vrstev. Každá z těchto vrstev pak „komunikuje“ s vrstvou nižší přes definované rozhraní.

#### Datová struktura

V rámci návrhu je také velmi podstatné navrhnout datový model databázového systému, který bude aplikace využívat k manipulaci s daty. Existují tři úrovně abstrakce, na základě kterých je datový model navržen: [10]

* **konceptuální –** určuje co je obsahem systému. Je potřeba popsat pouze obsah datové struktury nezávisle na konkrétním databázovém systému,
* **logická –** určuje jak je obsah systémů v dané technologii realizován. Na této úrovni se v relačních databázích používá tzv. relační schéma, které obsahuje tabulky včetně jejích sloupců, atributů, primárních a cizích klíčů,
* **fyzická –** určuje čím je technologické řešení realizováno. Vybírámekonkrétní databázovou platformu, ve které bude vytvořena a využíváme zde například její programovací jazyk.

Datový model je pak znázorněn například v tzv. entitně-relačním diagramu, zkráceně ER diagramu, kde se uvádí jednotlivé entity, obsahující název a atributy, vazby mezi těmito entitami a tzv. kardinalitu.

#### Desing

Při návrhu grafického uživatelského rozhraní, neboli GUI (Graphical User Interface) nejprve vytváříme tzv. drátěný model uživatelského rozhraní, neboli wireframe. Jde především o schématický návrh, ve kterém jsou potlačeny detaily vzhledu jednotlivých GUI komponent a jejich obsah. [8]

Na základě wireframů je vytvořen vizuální vzhled komponent ve formě prototypů.

Rozlišujeme dva druhy prototypů:

* **low fidelity** – je zjednodušený prototyp produktu, který slouží k přetvoření wireframes v subjekt vhodný pro otestování budoucím uživatelem, aby byly chyby odchyceny v raném stádiu,
* **high fidelity –** je funkční a interaktivní prototyp, který zahrnuje veškeré detaily a funkčnosti finálního produktu.

Prototypy jsou poté testovány a je získávána zpětná vazba od uživatelů. Poznatky z testů jsou pak následně v designu GUI opraveny a přejde se do fáze implementace.

### Implementace

V rámci této fáze dochází k realizaci prototypu a vytvoření aplikace. Obvykle tato fáze trvá nejdéle a během vytváření i po dokončení by měla být testována.

### Testování

Existuje mnoho druhů testování ať už s uživatelem nebo bez uživatele. Mezi ně patří například testy vývojářské, integrační, systémové nebo uživatelské akceptační testy. [9]

### Nasazení a údržba

Po akceptaci aplikace uživatelem následuje poslední fáze a tou je uvedení aplikace do testovacího či běžného provozu, předání dokumentace, uživatelské příručky a údržba.