Contents

1 Softwarový proces

Softwarový proces Aktivity, které je potřeba uskutečnit k tomu, aby software vznikl. Jedná se o jejich souslednost a opakování, o vstupy a výstupy jednotlivých aktivit a jejich nároky.

Model životního cyklu SW Viz. Softwarový proces.

Software process improvement Ladění a vylepšování procesu softwaru.

1.1 Nutné aktivity

- Specifikace - co bude systém dělat.
- Architektura a design - z jakých částí se systém bude skládat.
- Implementace - vlastní provedení systému.
- Validace - otestování systému.
- Další rozvoj - úpravy systému na základě měnících se požadavků.

1.2 Model životního cyklu - SDLC

1. Analýza požadavků
2. Specifikace požadavků
3. Vývoj software
4. Integrace a nasazení
5. Provoz a údržba
6. Zhodnocení výkonnosti

1.3 Modely vývoje

1.3.1 Vývoj dle plánu

- Aktivity jsou plánovány dopředu.
- Pokrok je měřen porovnáním stavu a plánu.
- Větší režie v případě změn.

Waterfall

- Oddělené fáze
  - Analýza požadavků
  - Design

  - Implementace
  - Testování
  - Provoz a údržba
- Výhody
  - Jasně definovaný plán
  - Predikovatelnost (čas, rozsah, cena)
  - Snadná koordinace práce
- Nevýhody
  - Nutné pochopení požadavů už na začátku
  - Reakce na změny
  - Rychlost dodávky
- Spolupráce se zákazníkem
  - V přesně definovaných okamžicích
  - Lze dop

1.3.2 Iterativní vývoj

- Oproti vodopádu:
  - Několik verzí systému
  - Jednotlivé verze se dělají vodopádem
- Výhody
  - Jasně definovaný plán
  - Predikovatelnost
  - Snadná koordinace
  - Zákazník má přístup k verzím/prototypům
- Nevýhody
  - Je nutné chápat co klient chce již na začátku

Spirálový model Iterace dělena na 4 kvadranty:

- plánování,
- odhad rizik,
- inženýrství,
- hodnocení.

1.3.3 MDA - Model Driven Architecture

- Využívá UML
- Model nezávislý na počítačovém zpracování (CIM)
- Model nezávislý na platformě (PIM)
- Model specifický pro platformu (PSM)
- Zdrojový kód aplikace (Implementace)

1.3.4 Agilní vývoj

- Plánování je jen po malých částech.
- Snadné změny směru v případě změny požadavků.

1.4 Různé pohledy na modely vývoje

1.4.1 Flexibilita

Flexibilita Reakce na změny. Určuje rychlost a finanční náklady na provedené změny.

- Vodopád
  - Nepružný
  - Vysoké náklady
- Iterativní
  - Lze zakomponovat změny do další iterace
  - Náklady nižší než u vodopádu
- Agilní
  - Snadné změny, jsou očekávané
  - Náklady velmi nízké

1.4.2 Predikovatelnost

Predikovatelnost Dopředu známá cena a datum dokončení.

- Vodopád
  - Velmi dobře předvídatelný, vývoj podle plánu
- Iterativní
  - Době předvídatelný, vývoj podle plánu
- Agilní
  - Těžko předvídatelný, plán jen na krátké období

1.4.3 Architektura a design

Architektura a design Správnost navržení systému, dodržování principů návrhu.

- Vodopád
  - Velmi dobrá architektura
- Iterativní
  - Velmi dobrá architektura
  - Možnost zanesení chyb v dalších iteracích
- Agilní
  - Často špatná architektura
  - Riziko zanesení problému v každé iteraci

1.4.4 Implementace

Prostor pro dodání kvalitního díla. Požadavky na programátory.

Quality assurance Zkráceně QA, týká se obecně všech procesů od návrhu po dokumentaci. Cílem je ujistit se, že je produkt vytvářen s odpovídající kvalitou.

- Vodopád
  - Kvalitní, hodně prostoru pro QA
  - Dodržují se coding standards, časté revize
- Iterativní
  - Dostatek prostoru pro QA
  - Revize, coding standards
  - Riziko zanesení problému v dalších iteracích
- Agilní
  - Nutný kvalitní tým
  - Riziko nekvalitní práce, pokud není čas na revize

1.4.5 Dokumentace

Dokumentace Dokument s informacemi o implementaci a nasazení systému.

- Vodopád
  - Kvalitní dokumentace
- Iterativní
  - Nutno dodržet napříč verzemi
- Agilní
  - Často nekvalitní
  - Obtížné udržet napříč sprinty

1.4.6 Spoluprace se zakaznikem

Součinnost zákazníka Jak moc a často se musí zákazník podílet na projektu.

- Vodopád a Iterativní vývoj
  - V přesně definovaných okamžicích
  - Lze dobře plánovat
- Agilní
  - Nutné v průběhu celého projektu
  - Riziko selhání, pokud spolupráce není dobrá

1.4.7 Smlouva na dodávku

Smliuva na dodávku Určuje jestli bude dílo dodáno za fixní cenu ve stanovený čas.

- Vodopád
  - Ano
- Iterativní
  - Ano, ošetřuje se rozsah verzí
- Agilní
  - Ne, požadavky nejsou známy dopředu
  - Aktuální modely nákupu tomu nejsou nakloněny

1.5 Fáze projektu

1.5.1 Primární činnosti

Tvoří hodnoty produktu.

- Modelování business modelu zákazníka
- Sběr požadavků
- Analýza a design
- Implementace
- Testování
- Nasazení

1.5.2 Podpůrné činnosti

Tvoří se stále po dobu běhu projektu. Zaobalují primární činnosti.

- Změnové a konfigurační řízení
- Projektové řízení
- Správa prostředí

2 Requirements engineering

Requirements engineering Proces, který pokrývá veškeré aktivity spojené s návrhem, dokumentací a údržbou požadavů na systém.

2.1 Schématický pohled - fáze

- Elicitation (schůzky, jednání, připomínkování, pozorování uživatelů, ...)
- Analysis (přemýšlení, vymýšlení, debaty a poznámky)
- Specification (dekompozice, psaní a používání notace)
- Verification (čtení textů, schůzkym jednání, poromítání GUI, rozsah, ...)

2.2 Význam procesu

- Špatně definované požadavky způsobují neúspěch projektů
- Specifikace poždavků znamená:
  - Zadání práce pro techniky
  - Specifikace toho, co bude dodáno
  - Základní část dokumentace
- Rozsah projetku je brán jako parametr ceny

2.3 Základní pojmy

Rozsah (scope) Množství práce (typicky včetně dodávky systému).

Nabídka Obsahuje definici rozsahu.

Specifikace požadavků Dokumenty obsahující požadavky na systém. Vytváří se až po nabídce.

2.4 Typy požadavků

Je potřeba myslet na všechny typy požadavků. Dotazy je potřeba směřovat na relevantní skupiny zainteresovaných osob.

- Požadavky na vlastní funkce a rozhranní
- Požadavky na rozhraní (uživatelské, SW, HW, grafické, komunikační, ...)
- Nefunkční požadavky (výkon, bezpečnost, spolehlivost, dostupnost, škálovatelnost)
- Ostaní požadavky
  - Legislativní
  - Vícejazyčnost
  - Technologie
  - Platforma

2.4.1 Požadavky na požadavky - dle IEEE

- Correct - Přesně popisuje chování systému.
- Unambiguous - Nemělo by možné si je různak vyložit.
- Complete - Obsahuje úplné požadavky na funkčnost systému a jeho vlastnosti.
- Consistent - Požadavky se navzájem nevylučují.
- Ranked - Třídění podle důležitosti. Každý požadavek je řazen podle důležitosti a kritičnosti.
- Verifiable - Musí být možné ověřit, že byl požadavek naplněn.
- Modifiable - Je možné požadavek upravit.
- Traceable - Sledování změn požadavků. Je nutné mít zaznamenanou každou změnu.

3 Architektura a design

Software architecture Popisuje programovací paradigmata, architektonické styly, principy a standardy. Definuje komponenty a vztahy mezi nimi.

Software design Realizace funkčních požadavků. Jedná se o návrhové vzory, programovací idiomy a refaktoring. Společně s architekturou udává postup implementace.

Business process architecture Zaměřuje se na obchodní strategie, řídící procesy a organizační struktury.

Information technology architecture Zkoumá HW a SW infrastrukturu organizace nutnou pro chod systému. Umožňuje identifikovat HW a SW požadavky.

Information architecture Zkoumá uložiště a databáze zpracovávaných dat, jakým způsobem jsou ukládána, používána a zpracovávána.

3.1 Design

S designem jsou spjaté koncepty:

- dekompozice,
- abstrakce,
- zapouzdření,
- koheze,
- vazby.

Pojmy spojené s designem:

- abstraktní datový typ,
- typ,
- třída,
- objekt,
- instance,
- modul.

3.2 Architektonické styly

Popisují způsobym jak navrhovat moduly a komunikaci mezi nimi. Mezi tyto styly patří:

- MVC - Model-View-Controller,
- MVP - Model-View-Presenter,
- Event driven architecture,
- Layered architecture,
- Repositories.

3.3 Pohled na architekturu

Logický Poskytuje abstraktní pohled na problém. Obsahuje diagramy tříd, ale neobsahuje implementaci.

Modulární (Development) Člení problém do modulů a subsytémů. Definuje jak postupovat při implementaci (jazyk, architekturu,..).

Koordinační (Process) Zohledňuje spolupráci a synchronizaci procesů. Definuje výkon, dostupnost, toleranci výpadku a integritu.

Fyzický Definuje škálovatelnost, výkon a dostupnost. Obsahuje popis mapování SW na HW. Snaha o nízký dopad na zdrojové kódy.

3.4 Frameworky

- Znovupoužitelney návrh pro SW systém
- Základ při vývoji jiných SW aplikací
- Diktuje architekturu systému

3.5 Integrace

- Spojeno s tématikou enterprise architektury
- Netechnologické (procesy, entity)
- Používání buzzwords (EAI, SOA, MOM)

3.6 Shared database

- Více aplikací sdílí společnou databázi
- Odpadají problémy se synchronizací
- Je ale potřeba vytvořit unifikované schéma pro všechny aplikace
- Úzké hrdlo výkonnosti

3.7 Remote procedure call

- Aplikace vlastní data a stará se o jejich integritu, ostatní aplikace volají funkce, které nabízí
- Mnoho technologií (JAVA RMI, .NET, Webové služby, ...)

3.8 Cloud

- IaaS - Infrasctructure as service
- PaaS - Platform as a service
- SaaS - Software as a service

4 Návrhové vzory

4.1 Service Oriented Architecture - SOA

4.2 Operational Data Store - ODS

5 Design and Construction

5.1 Návrhové vzory

5.1.1 Principy

DRY Don't repeat yourself. Zamezit co nejvíce duplicitám v kódu.

SRP Single responsibility principle. Každý logický celek dělá jednu věc.

5.1.2 Test Drive Development

- Implemetuje se na základě rozhraní vytvořeného v testech.
- Produkuje testovatelný SW
- Zajišťuje existenci testů

5.1.3 Self-documenting code

- Svou strukturou a formou omezuje nutnost číst dokumentaci
- Nezajišťuje úplnou absenci dokumentace

6 Testování

- Zkoušení/simulace provozu
- Analýza s cílem nalézt chyby SW
- Zhodnocení atributů schopností SW

Test plan Definuje strategii testů, obsahuje test coverage, test methods a responsibilities.

Test case Množina podmínek, za kterých tester určí, zda aplikace funguje správně.

Test script Množina instrukcí, které budou během testování provedeny.

Test data Data speciálně identifikovaná pro využití v rámci testování.

Test report Výsledek jednoho či více testů.

6.1 Principy

- Kompletní testování není možná
- Měření a sledování je důležité

6.2 Typy testů

6.2.1 Developer

Testování samotným vývojářem před přidáním do Unit testů.

6.2.2 Unit testy

Testy základních celků (třídy a metody).

6.2.3 Integrační testy

- Komunikace mezi jednotlivými systémy.
- Test integrace základních celků do systému.

6.2.4 Smoke testy

Spouštějí se v okamžiku, kdy je dokončen vývoj aplikace a lze ji spustit. Pouští se po integračních testech.

- Testuje implementaci, instalaci a spuštění jednotlivých částí systému.
- Zaměřuje se na hlavní části systému.
- Pokud jsou testy splněny, přechází se k systémovému testování.
- U menších projektů kde nejsou ostatní testy se provádí většinou pouze *smoke testy*.

6.2.5 Systémové

Testování aplikace jako celku. Testují se všechny části spolu s dodávaným HW. Obsahuje test nároků na HW.

6.2.6 Kvalifikační

Kontrola splnění požadavů na SW. Odehrávají se u dodavatele před dodávkou. Pokud SW projde je dodán zákazníkovi.

6.2.7 Akceptační

Obdoba *kvalifikačních testů*, které se odehrávají u zákazníka. Provádí se podle připravených scénářů vytvořených ve spolupráci s dodavatelem.

6.2.8 Regresní testy

Testují, zda změny SW (např. nové funkce) nenarušily funkcionalitu stávajících částí, na které by změny neměly mít vliv.

6.3 Testovací techniky

6.3.1 Black box testy

- Testuje se oproti rozhraní
- Implementace není známá
- Není nutné často upravovat

6.3.2 White box

- Strukturované testy
- Přihlíží se k implementaci
- Změna implementaci často testy rozbije

6.4 Požadavky na testy

- Najít chyby aby byly odstraněny
- Rozhodnout se jestli aplikaci nasadit
- Minimalizovat náklady na podporu
- Měření kvality

7 Quality assurance

Kvalita Souhrn vlastností nebo charakteristik služby, které souvisí se schopností splnit předpokládané potřeby.

Quality assurance Množina aktivit, jejichž cílem je zajistit kvalitu. Nedokáže zajistit kvalitu, ale výrazně zvyšuje pravděpodobnost, že se tak stane.

Validace, verifikace Množnia aktivit, s cílem zjistit zda daný artefakt splňuje nároky kladené na něj. Je to snaha o ověření správnosti fungování programu.

Testování Proces/množina aktivit s cílem změřit kvalitu vytvářeného SW.

7.1 V-model

- Slouží pro zajištění kvality.
- Na každé úrovni pyramidy se dělá review.
- Při tvorbě plánu se tvoří kritéria pro odpovídající testy.

8 Configuration management

Softwarový produkt Úplný soubor počítačových programů, postupů, související dokumentace, určený pro dodání uživateli.

Softwarová položka Jakákoliv identifikovaná část softwarového produktu v průběžném nebo konečném stadiu vývoje.

Konfigurační řízení Zajištění plného řízení konfigurace SW produktu a související dokumentace v průběhu životního cyklu.

Změnové řízení Součást konfiguračního řízení. Je to řízení rozsahu nad rámec původně domluveného rozsahu.

8.1 Řízení verzí

SCM/VCM Verzovací systém (např. *git*, *svn*, ...)

Konfigurační jednotka Jeden soubor s konfigurací systému.

- Identifikace elementů a verzí.
- Evidence změn během stádia vývoje.
- Umožňuje paralelní práci.
- Ozačování verzí pomocí *tagů* a *verzí*.

8.2 Změnové řízení

- Využívá se *issue tracker*, např. Bugzilla.
- Měří se čas, který změny zabraly z důvodu nacenění.

8.3 Release management

Integrační platforma *Cruise control* Každodenní kontrola zdrojových kódů. Umožňuje rychlou zpětnou vazbu v případě problémů.

Vytváří se každodenní build, deploy a test.

8.4 Průběžná integrace / CI

Umožňuje rychlé nasazení, spuštění automatických testů, automatizuje kompilaci, automatizuje code review. Vyžaduje, aby tým pracoval s nejaktuálnější možnou verzí projektu.

9 Vývojové prostředí

DevOps Zkratka z Development and Operations. Forma vývojového cyklu, kde je snaha o vysokou automatizaci.

Přestože je systém primárně určený pro cílové prostředí u zákazníka, je nutné zavédst i další prostředí.

9.1 Prostředí u dodavatele

- Vývojové
- Integrační
- Testovací

9.2 U zákazníka

- Akceptační
- Více produkčních prostředí

9.3 Developmnet Operations

- Každý krok je maximálně automatizovaný
- Vše je verzováno i testováno (zdorojové kódy, databáze, data)
- Po všechna prostředí se používá unifikovaný proces
- Podporuje agilní vývoj

Efektivně se využívá nástrojů jako je git, CI/CD, build automation. Provádí se satická analýza a automatizované testy.

Hlavním přínosem je snížení TTM business požadavků, množství chyb a nákladů na zdroje.

10 Projektové řízení

Projekt Dočasná organizační jednotka, která ze vytvořena s cílem doručit jeden nebo více produktů podle obchodního plánu.

10.1 Charakteristiky projektu

- Proměnlivý
- Dočasný
- Unikátní
- Nejistý

10.2 Problémy IT projektů

- Nerealistické termíny
- Změny v rozsahu
- Špatné řízení rizik
- Špatná komunakice
- Viditelnost pokroku
- Špatně definovaná vize a cíle projektu

10.3 Aspekty projektu

- Náklady
- Čas
- Rozsah

10.4 Metodiky

10.4.1 Prince2

Obecná metodika pro řízení projektů (nejen v IT). Stojí na sedmi principech:

- Business justification - Jak projekt nebo úkol splňují své cíle
- Learn from experience
- Role a odpovědnosti
- Manage by stages - Více release a iterací je jedna stage
- Manage by exception - Řízení zaměřené na identifikaci a řešení situací vyhýbající se normálu
- Zaměření na produkty
- Přizpůsobení prostředí projektu (velikosti, složitosti, důležitosti, ...)

10.5 Projektový manažer

Udržuje plán projektu aktuální, stará se o:

- Termíny
- Závazky
- Zdroje
- Rizika
- Běh projektu

Pracuje pro lidi na projektu a reportuje stav jak své firmě, tak zákazníkovi. Jeho náplní je odstínit tým od nepříjemností. Zodpovídá za to, co dělají členové týmu. Rozděluje úkoly a kontroluje jejich splnění.

10.5.1 Self management

- Je důležité zvládnout organizeci vlastního času
- Úkoly do jedné minuty plnit ihned
- Rozlišovat mezi důležitými a urgentními úkoly

10.6 Nástroje projektového manažera

- Plán projektu, WBS
  - Menší úkoly na 1-5MD
  - Měří aktuální stav projektu
  - Sleduje earned values
- Výkazy
  - Hlídání odvedené práce
  - Kontrola zbývající práce
- Nabídka
  - Cenotvorba a termíny
  - Odhady a předpoklady, zdroje, milníky, harmonogram, ...

10.7 Řízení rizik

Riziko Ohrožení projektu, ceny, termínu, kvality nebo jiné vlastnosti projektu. Může se jednat o ohrožení *Business case*.

10.8 Měření a metriky

Projektové metriky Sloiží k taktickým účelům, odhadují čas, náklady a monitorují projekt.

Metriky orientované na velikost Odvozené z velikosti produktu a normalizovaná faktorem efektivity (např. počet řádků, počet stránek v dokumentaci).

Funkčně orientované metriky Měří vztah mezi využitelností informační domény a složitostí systému (např. počet databázových dotazů určuje nutný výkon serveru).

Na projektu lze měřit čas, snahu, kvalitu a rozsah. U softwaru lze pak měřit počet defektů, produktivita a efektivita testů.

10.8.1 Kategorie

- Čas - z evidence práce.
- Velikost - počet obrazovek, řádků kód nebo tříd.
- Pracnost - člověkohodiny.
- Kvalita - počet chyb v issue trackeru.

10.8.2 Využití

- Vhodné pro historii projektů - lze lépe odhadovat časovou pracnost
- Vytváření nových nabídek na základě metrik a servisních smluv
- Vhodné pro argumentaci o výši cen

10.9 Historie

10.10 Milníky

10.10.1 LCO - Lifecycle Objectives

- Milník na konci počáteční fáze
- Jsou známy požadavky klienta
- Kontrolují se cíle projektu
- Rozhoduje se o uskutečnění projektu
- Kritéria hodnocení:
  - Identifikovány všechny zainteresované strany
  - Je znám rozsah, cena a plán
  - Odsouhlasen rozsah, náklady harmonogram, priority a rizika

10.10.2 LCA - Lifecycle Architecture

- Milník na konci fáze zpracování
- Jsou zanalyzovány a specifiovány požadavky klienta
- Kontroluje se rozsah, cíl, výběr architektury a řešení rizik
- Hodnotí se:
  - Stanoveny testovací přístupy
  - Vyhodnocení spustitelných prototypů dokazuje, že všechna hlavní rizika jsou vhodně ošetřena
  - Jsou akceptované aktuální a plánované náklady

10.10.3 IOC - Initial operational capability

- Milník na konci fáze konstrukce
- Předchází fázi transaction (přechod)
- Systém je funkční a prošel fází *alfa testování*
- Do systému je vpuštěn zákazník a uživatelé
- Existuje příručka s popisem aktuálního stavu
- Systém je nasazen, ale není ve finální fázi (testování, ladění, ...)
- Odhaduhe se čas nasazení do testovacího režimu (testovací provoz v reálné zátěži)
- Hodnotí se:
  - Verze je stabilní a je možné ji nasadit mezi uživatele
  - Aktuální a plánované náklady jsou akceptované

11 Údržba

Údržba software Systematický rozvoj produktu s aplikací drobných změn. Během údržby se také odstraňují chyby a problémy.

- Systém byl dodán v rozsahu dle nabídky
- Systém byl akceptován a rutinně provozován
- Systém neobsahoval příliš mnoho chyb

11.1 Dokumentace

Využívá se jako komunikační prostředek. Obsahuje záznamy o domluvách a dohodách. Primárně slouží jako zdroj informací o implementaci systému. Zajišťuje udržitelnost software, potencionálně dlouho po vytvoření. Je potřeba mít specifikaci k uhlídání rozsahu. Znát architekturu aby bylo možné ji dodržet. Jednotlivé změny je také potřeba zdokumentovat.

11.1.1 Typy dokumentace

- Uživatelská
  - Úvodní manuál
  - Popis funkcí
  - Systémový manuál
  - Instalační příručka
  - Administrační příručka
- Systémová
  - Požadavky na systém
  - Specifikace, architektura, design
  - Zdrojový kód s komentářit
  - Dokumenty údržby (známé chyby, závislost na infrastruktuře)

11.2 Typy údržby

Směrnice ISO/IEC 14764 definuje následující typy údržby:

Corrective Za účelem opravy nalezených chyb a problémů.

Adaptive Za účelem udržení použitelnosti SW v měnícím se prostředí.

Perfective Za účelem zlepšení výkonnosti nebo udržovatelnosti.

Preventive Za účelem detekce a opravy latentních chyb než se stanou skutečné.

11.3 SW development life cycle

Používá se *Miniwaterfall*. O systému má dodávající velkou znalost a změny jsou typicky malého rozsahu. Velmi efektivní způsob.

11.4 Měření

Velmi snadno se získávají přesná čísla. Měření slouží jako podklad pro servisní smlouvu na další léta.

11.5 Vývojové prostředí

Využívá se prostředí podobné produkci, které se i podobně používá. Využívá se CI a smoked tests.

11.6 Architektura

Architektura musí být navržena tak, aby byla schopna absorbovat nové požadavky.

11.7 CM

Evidují se všechny požadavky zákazníka. Je striktně proces změnového řízení.

11.8 Testy

Komplexní systémy se většinou po každé změně netestují celé, buď se netestují vůbec nebo:

- existují regresní automatické testy,
- průběžně se kontroluje každá chyba,
- testy jsou naplánované a zorganizované,
- existuje záznam o testování,
- existují akceptační testy a jejich záznamy.

11.9 Tým

Tým údržby je založen na lidech, kteří ho prvotně vyvíjeli. Z důvodu únavy se lidé obměňují až se vytvoří úplně nový tým. Nové členy je tedy potřeba řádně zaučit.

11.10 Ekonomika

Cílem je, aby údržba byla výdělečná. Je potřeba mít efektivní proces a velmi přesné odhady.

11.11 Odhady

Je potřeba udělat mnoho odhadů. Při odhadech je nutné být konzistentní, ideálně používat stále stejnou metodiku odhadů. Odchylky je potřeba umět zdůvodnit.

11.12 Technická podpora

Technická podpora Služba zajišťující pomoc uživatelům při řešení chyb.

Rozlišuje se na 3 úrovně: L1, L2 a L3.

L1 *First-line-support*. Styčný bod mezi uživateli a techniky. Náplní práce je radit uživatelům s obsluhou systému a přijímání chybových hlášení.

L2 Administrátorská podpora. Technici řeší incidenty systému - zpravidla ještě před náhlášením díky monitoringu.

L3 Expertní podpora. Technici mají expertní znalost systému - často přímo vývojáři.

11.12.1 Placená podpora

Z pohledu zákazníka:

- Zajištění opravy chyb i po skončení záruky
- Minimalizace nedostupnosti služeb

Z pohledu dodavatele:

- Zajištění příjmu
- Pokud firma vyvíjí kvalitní SW, lze zpravidla efektivně poskytovat
- Možnost snáze reagovat na změnové požadavky (technici mají detailní znalost systému)

11.12.2 SLA - Service level agreement

V rámci provozu systému a jeho podpory jsou garantovány určité parametry. Za nesplnění nebo vybočení z domluvených mezí jsou udělovány sankce.

Typické parametry jsou:

- dostupnost,
- doba reakce,
- doba za kterou musí být chyba vyřešena.

11.12.3 Rozdělení

Podle času:

- 24/7 - nepřetržitá,
- 8x5 - pouze v pracovní době,
- 10x5 - rozšířená pracovní doba.

Podle intenzity:

- On-site - dostupná přímo u zákazníka. Typicky L1.
- On-call - podpora po telefonu (levnější varianta).