KAE/ASE
Aplikovaný SW pro elektroniku

Ing. Petr Weissar, Ph.D.

verze 2018/19

Program KAE/ASE - přednášky

1. Úvodní přednáška
   - C++, C# v MS VisualStudio
   - Základní DB – použitelné např. pro ukládaní dat
2. WebService a související technologie (tenký klient – prohlížeč)
3. Komunikace sockety, TCP/IP
4. Tenký klient – základní technologie
5. Prezentace dat + .NET on ARM
   - Datové a souborové formáty, využití tlustý/tenký klient
   - Micro.NET Framework
6. Přenosy dat
   - Bezpečnost přenosu dat, řešení problémů
   - Sériový port
   - Bezdrátové přenosy - lokální
7. Embedded systémy
   - Real-time OS – aspekty
   - PDA a přesah do současných zařízení
   - Linux a vazba na HW
   - RFID
   - GPS
8. Rozšiřující karty PC – ISA, PCI, PCIe, PC-Card
9. Populární zkratky - RFID, GPS, mobilní přenosy, ...
10. Cloud – služby, ukládání dat, vizualizace, IoT
11. Tenký klient – ukázka konstrukcí a použitelných objektů
12. Android – ukázka konstrukcí a použitelných objektů v C#
13. Témata na přání, rezerva

Přednášky (tyto slide) a další informace najdete v CW na **http://portal.zcu.cz**

KAE/ASE - cvičení

1 Úvod + databáze
   Opakování základních kroků v programování C#
   Připojení k DB
2 WebService
3 Komunikace sockety – UDP
4 Tenký klient – web aplikace pomocí WebForms
5 Vizualizace dat – lokálně, www – pokračování
6 Micro.NET framework – HW a základní aplikace ve VS
7 Micro.NET framework – další možnosti HW
8 Micro.NET framework – komunikace
9 Intel Galileo – Embedded Linux platforma
10 Cloudové služby – příklad použití
11 Bezdrátová komunikace – WiFi modul připojený přes UART
12 Semestrální práce – jednoduchá app posílající data do PC, přenos do centrální DB, zobrazení dat
13 Rezerva/Zápočet

Programy řešené na cvičení a informace na portálu ZČU (**CourseWare**)

1. přednáška

- Programování pro moderní OS
- .NET Framework, MS Visual Studio
- C++, C#

- Relační databáze – pojmy
- SQL jazyk
- Relace mezi tabulkami, normalizace
- Databázové stroje
- Programové řešení

Cílové platformy aplikací

- Běžná (stolní) PC
  - Důraz na komfort a efektivitu práce
  - Dostatek výkonu, paměti, možnosti interakce
  - Efektivně Linux, Win32/64, OS X

- Přenosné počítače a zařízení
  - Důraz především na spotřebu
  - Jiná filozofie ovládání (nyní dotykové)
  - Zpočátku relativně malý výkon (PDA začínaly minimálně Intel PXA255 – 400MHz, 64+MB RAM, 64+MB Flash)
  - Windows CE/Win Phone, příp. PalmOS, Android, iOS
- Průmyslové počítače
  - Výkon od x51 po výkonné P4, Core2Duo, iX, …
  - Důraz především na spolehlivost – větráky a disky
  - Speciální systémy nebo upravené „normální" OS

Tlustý vs. tenký klient

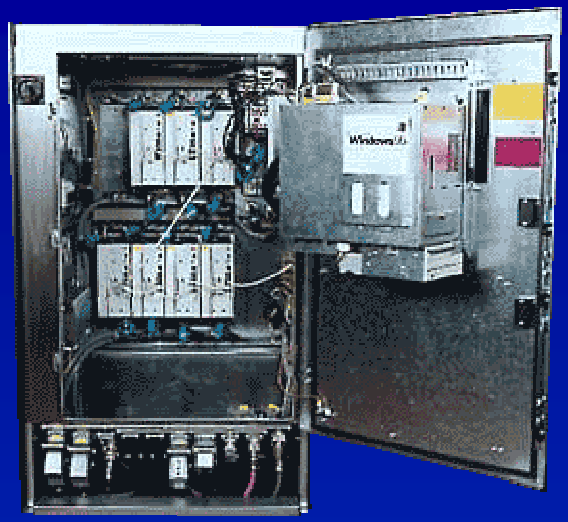
- Velká část aplikací je zpracování a sdílení dat

- **Tlustý klient** zpracovává většinu informací na lokálním počítači, příp. si data pro zpracování „natáhne“ ze vzdáleného úložiště (databáze apod.)
- **Tenký klient** typicky využívá na lokálním počítači uživatelské vstupy a výstupy, datové operace a logiku práce zpracovává vzdálený server – typicky stačí WWW prohlížeč
- Zjednodušeně: Tlustý klient potřebuje výkon na lokálním počítači, tenký klient potřebuje konektivitu na výkonný server

Použití OS

Embedded OS

+ Ultra small footprint
+ Range of Processors
+ Mobile Applications
+ Secure function
++ Real-time OS



Windows XP Embedded
Windows 7 Embedded
Windows 8 Embedded
Windows 10 IoT

+ PC Architecture
+ Full connectivity
+ Full Win32 capacity
+ Soft Real-time

Vizuální operační systémy

- Microsoft Windows
  - Win16 – Windows 3.1, 3.11
  - Win32 – 95, 98, Me, 2000, XP, Vista
  - .NET – nativně od Server2003, Vista (dříve Longhorn)
  - Windows Server 2008 (R2), 2012 (R2)
  - Windows 7+
- Linux
  - různé Window Managery nad X-Window jádrem
- Apple
  - OS X – na bázi Unixu + výkonné grafické prostředí Aqua

Událostní programování

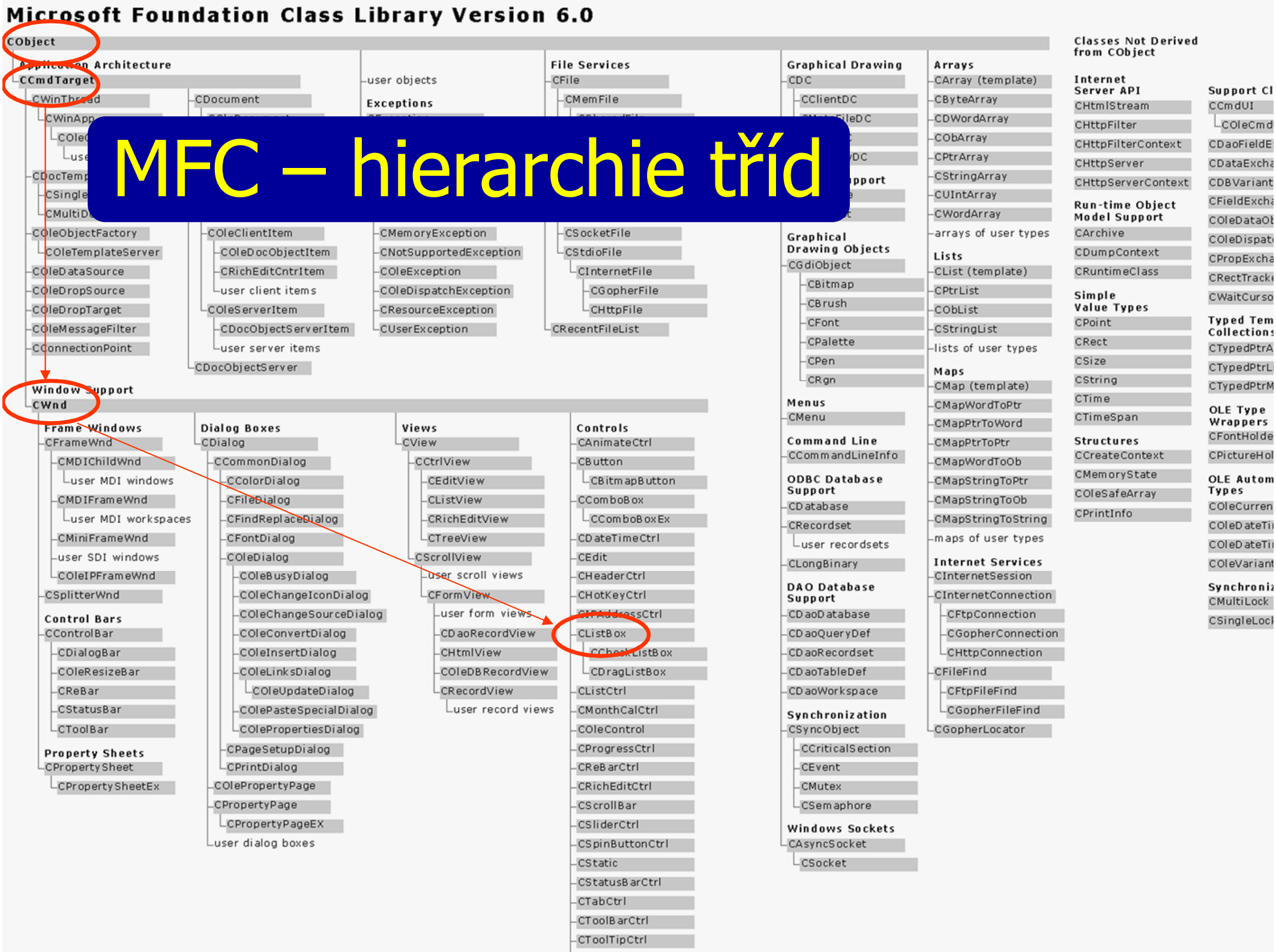
- Program neobsahuje smyčku **`while(!konec)`**
- Jedna funkce vyhrazena jako spouštěcí bod (obdoba **`main()`**)
- Dále funkce pro obsluhu událostí
  - stisknutí klávesy, klik na tlačítko (control) dialogu, klik do okna, událost od systémového časovače, událost od systému souborů, sériového kanálu či dalších I/O systémů, konec aplikace
- Obsluha se provede a řízení se vrací zpět systému
- Systém podle potřeby vyvolává příslušné funkce na obsluhu události

Stavební kameny včetně UI

- Grafická reprezentace interaktivních částí aplikace realizována službami systému (okna, prvky dialogů apod.)
- Systém odstiňuje před konkrétním HW
- Pro práci se systémem k dispozici balík funkcí – API (= Application Programming Interface)
- API pokrývá kromě UI také práci se soubory, sítí, standardizuje další HW např. na úroveň souborovou (sériový kanál)

Objektový pohled na UI

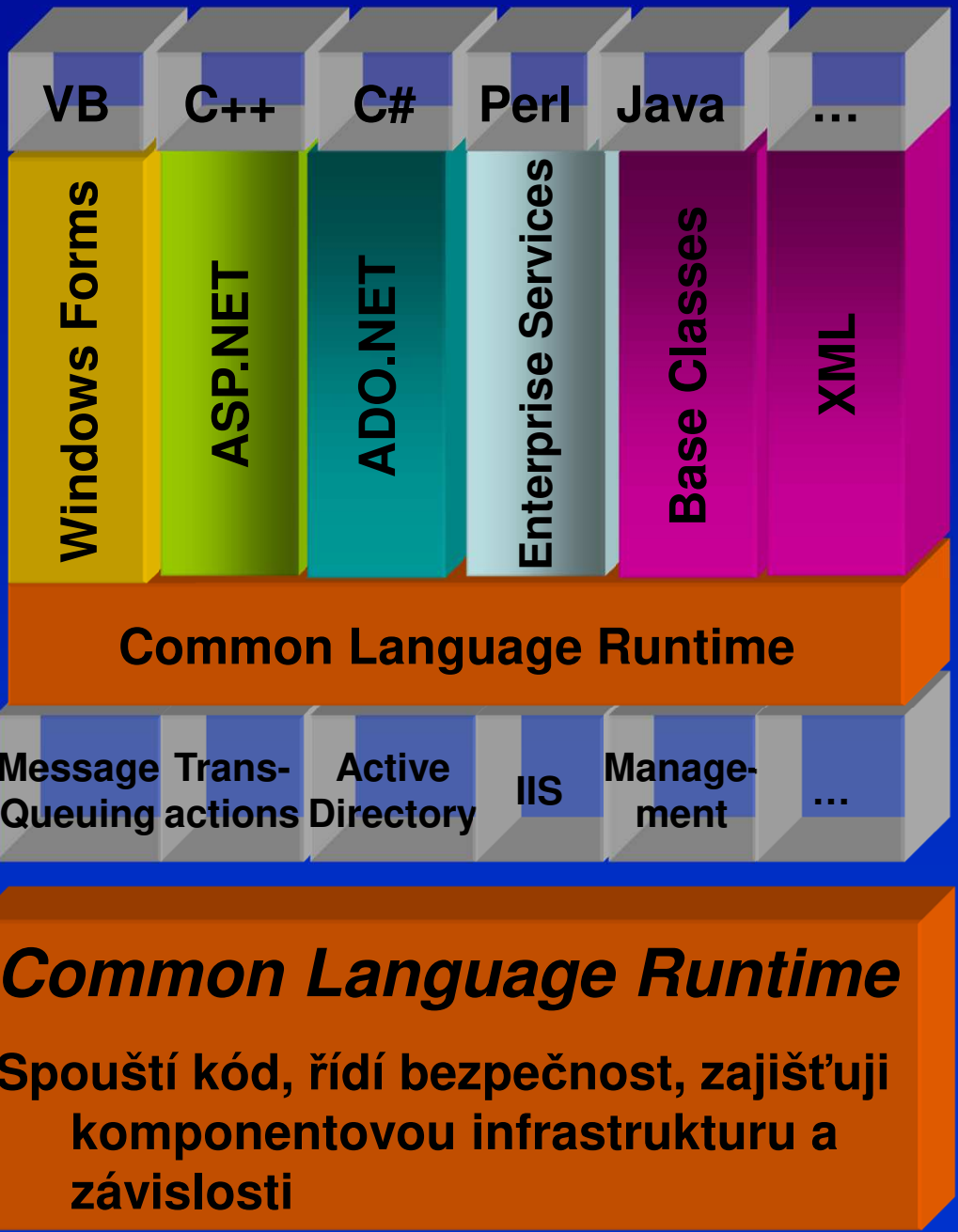
- První čistě objektový OS – NextSTEP
- Současné OS
  - interně zpravidla bez objektů – jazyk C a assembler (rychlost !)
  - objektové nadstavby různých programovacích jazyků a nástrojů zapouzdřují především vizuální prvky do hierarchie objektů (C++ apod.) a zároveň je podporováno vývojovým prostředím – IDE (Integrated Development Environment)
    - Microsoft Foundation Classes (MFC)
    - Object Windows Library (OWL)
    - VCL (Delphi ?) a další


Microsoft Foundation Class Library Version 6.0
CObject
Application Architecture
CCmdTarget
CWinThread
CWinApp
CDocTemplate
CSingleDocTemplate
CMultiDocTemplate
COleObjectFactory
COleTemplateServer
COleDataSource
COleDropSource
COleDropTarget
COleMessageFilter
CConnectionPoint
CDocument
COleClientItem
COleDocObjectItem
CRichEditCntrItem
user client items
COleServerItem
CDocObjectServerItem
user server items
CDocObjectServer
user objects
Exceptions
CMemoryException
CNotSupportedException
COleException
COleDispatchException
CResourceException
CUserException
File Services
CFile
CMemFile
CSocketFile
CStdioFile
CInternetFile
CGopherFile
CHttpFile
CRecentFileList
Graphical Drawing
CDC
CClientDC
Graphical Drawing Objects
CGdiObject
CBitmap
CBrush
CFont
CPalette
CPen
CRgn
Menus
CMenu
Command Line
CCommandLineInfo
ODBC Database Support
CDatabase
CRecordset
user recordsets
CLongBinary
DAO Database Support
CDaoDatabase
CDaoQueryDef
CDaoRecordset
CDaoTableDef
CDaoWorkspace
Synchronization
CSyncObject
CCriticalSection
CEvent
CMutex
CSemaphore
Windows Sockets
CAsyncSocket
CSocket
Arrays
CArray (template)
CByteArray
CDWordArray
CObArray
CPtrArray
CStringArray
CUIntArray
CWordArray
arrays of user types
Lists
CList (template)
CPtrList
CObList
CStringList
lists of user types
Maps
CMap (template)
CMapWordToPtr
CMapPtrToWord
CMapPtrToPtr
CMapWordToOb
CMapStringToPtr
CMapStringToOb
CMapStringToString
maps of user types
Internet Services
CInternetSession
CInternetConnection
CFtpConnection
CGopherConnection
CHttpConnection
CFileFind
CFtpFileFind
CGopherFileFind
CGopherLocator
Classes Not Derived from CObject
Internet Server API
CHtmlStream
CHttpFilter
CHttpFilterContext
CHttpServer
CHttpServerContext
Run-time Object Model Support
CArchive
CDumpContext
CRuntimeClass
Simple Value Types
CPoint
CRect
CSize
CString
CTime
CTimeSpan
Structures
CCreateContext
CMemoryState
COleSafeArray
CPrintInfo
Support Cl
CCmdUI
COleCmd
CDaoFieldE
CDataExcha
CDBVariant
CFieldExcha
COleDataOb
COleDispat
CPropExcha
CRectTracke
CWaitCurso
Typed Tem Collections
CTypedPtrA
CTypedPtrL
CTypedPtrM
OLE Type Wrappers
CFontHolde
CPictureHol
OLE Autom Types
COleCurren
COleDateTi
COleDateTi
COleVariant
Synchroni
CMultiLock
CSingleLock
Window Support
CWnd
Frame Windows
CFrameWnd
CMDIChildWnd
user MDI windows
CMDIFrameWnd
user MDI workspaces
CMiniFrameWnd
user SDI windows
COleIPFrameWnd
CSplitterWnd
Control Bars
CControlBar
CDialogBar
COleResizeBar
CReBar
CStatusBar
CToolBar
Property Sheets
CPropertySheet
CPropertySheetEx
Dialog Boxes
CDialog
CCommonDialog
CColorDialog
CFileDialog
CFindReplaceDialog
CFontDialog
COleDialog
COleBusyDialog
COleChangeIconDialog
COleChangeSourceDialog
COleConvertDialog
COleInsertDialog
COleLinksDialog
COleUpdateDialog
COlePasteSpecialDialog
COlePropertiesDialog
CPageSetupDialog
CPrintDialog
COlePropertyPage
CPropertyPage
CPropertyPageEX
user dialog boxes
Views
CView
CCtrlView
CEditView
CListView
CRichEditView
CTreeView
CScrollView
user scroll views
CFormView
user form views
CDaoRecordView
CHtmlView
COleDBRecordView
CRecordView
user record views
Controls
CAnimateCtrl
CButton
CBitmapButton
CComboBox
CComboBoxEx
CDateTimeCtrl
CEdit
CHeaderCtrl
CHotKeyCtrl
CIPAddressCtrl
CListBox
CCheckListBox
CDragListBox
CListCtrl
CMonthCalCtrl
COleControl
CProgressCtrl
CReBarCtrl
CRichEditCtrl
CScrollBar
CSliderCtrl
CSpinButtonCtrl
CStatic
CStatusBarCtrl
CTabCtrl
CToolBarCtrl
CToolTipCtrl
MFC – hierarchie tříd

.NET platforma

- Není to jen další objektová nadstavba na Win32 API
- Kompletní programové a programátorské prostředí:
  - Možno psát v libovolné jazyce (C#, Visual Basic, C++, Cobol, …)
  - Překlad do společného „mezikódu" – inspirace byte-code v Javě
  - Za běhu překlad do nativního kódu platformy – max. rychlost
- Elegantně podpořeno vývojovým prostředím – Visual Studio .NET
  - V průmyslu někde ještě verze 2003 - .NET 1.1
  - Vyjímečně v2005 (= Whidbey) - .NET 2.0
  - Stále užívaná v2008 (= Orcas) - .NET 3.5 (jádro 2.0 + rozšíření)
  - verze 2010 dostupná od roku 2010
    - .NET 4.0 – řada výrazných změn, slušná zpětná kompatibilita
    - provázání na nové technologie (Windows Azure apod.)
  - verze 2012 – obsahuje .NET 4.5
  - verze 2013 – přináší .NET 4.5.1 (pro Win 8.1)
  - verze 2015 – výhled k .NET 5.x
  - verze 2017 – používáme ve výuce, Community Edition volně k užití

Kam patří v .NET Framework



Windows Forms

Třídy pro bezpečné, jednoduše nasaditelné tlusté klienty

ASP.NET

Třídy a engine pro vytváření, nasazení a běh Web aplikací a služeb

ADO.NET

Třídy pro volně vázaný přístup k datům

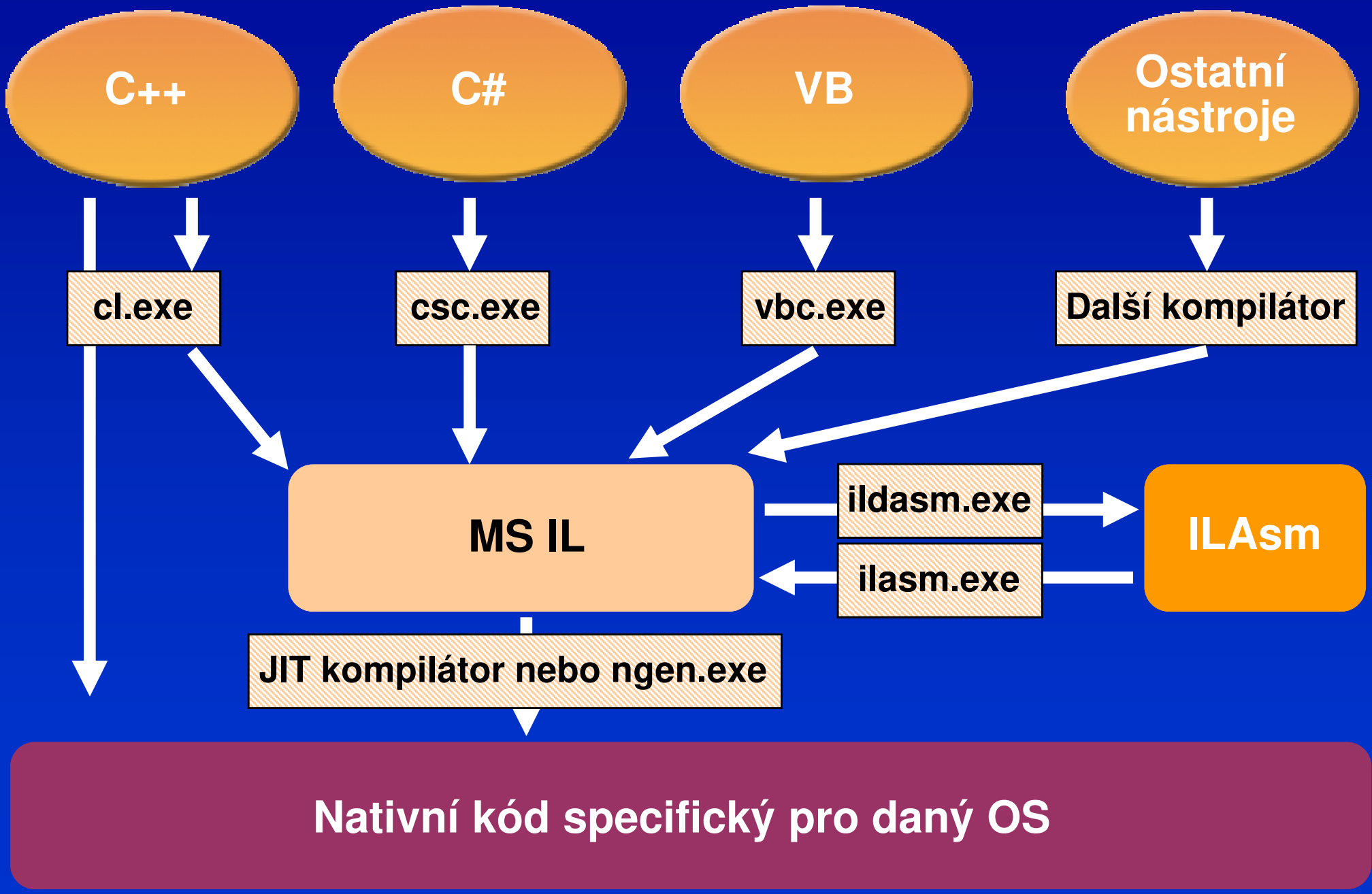
Enterprise Services

Transakce, object pooling, queued components, loosely coupled events, …

Common Language Runtime

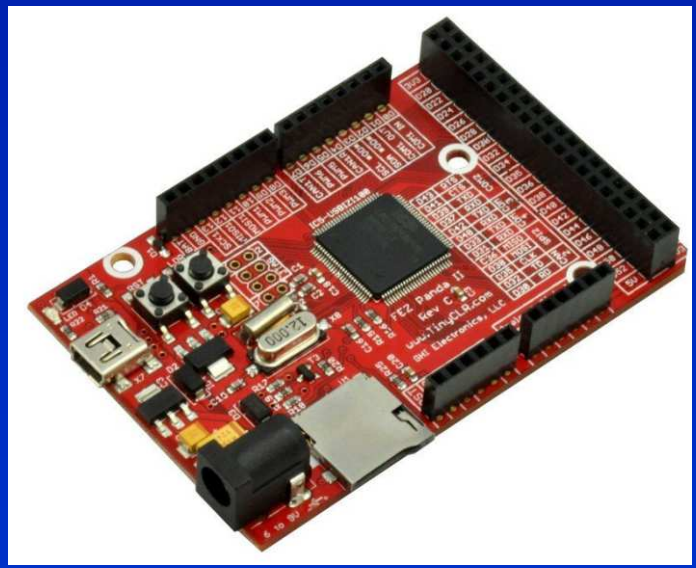
Spouští kód, řídí bezpečnost, zajišťuji komponentovou infrastrukturu a závislosti

Struktura překladu



.NET a jiné platformy

- CLI a C# standardizovány ECMA
- FreeBSD vzorová implementace (MS)
- Micro .NET Framework (MS)
  - pro malá zařízení
  - min. ARM7
  - min. 256k RAM, 512k FLASH
  - vyzkoušíme na cvičení
- ROTOR - iniciativa MS
  - zdrojové kódy implementace ECMA CLI a C# (WinXP)



.NET a jiné platformy - II

- Projekt Mono (www.mono.org)
  - primárně Linuxová platforma, existuje i Windows port
  - původně podporován např. firmou Novel
  - implementována velká část .NET 1.1 a většina prvků z 2.0 - především z ASP.NET, nyní též vybrané třídy z .NET 3.5
  - vývojové prostředí MonoDevelop, SharpDevelop, …
  - velká část vývoje ve firmě Xamarin
- Mono for Android (dříve MonoDroid)
  - novinka 2010 – port pro OS Android, původně Novell
  - 2011 – firma Xamarin
    - komerční ($299/licence nebo $999 s podporou Visual Studio)
    - po dokoupení MS je od 2016 součástí VS2015
- Mono for iOS (dříve MonoTouch)
  - Dostupné od 2010, nyní také Xamarin, komerční
  - .NET FW pro iOS (iPhone/iPad), umožňuje využít nativní prvky Apple OS

C++

- Největší počet použitelných platforem
- Objektový jazyk
  - zobecňuje struktury na třídy
  - členská data a členské funkce
  - omezení viditelnosti - public, protected, private
  - dědičnost (vlastností) - možno z více předků
- Zpětně "prakticky" kompatibilní s ANSI C
  - zůstávají staré nectnosti
    - ukazatele a ochrana paměti (neexistující)
    - složitý kód - velká zátěž překladače
  - max. využití HW
    - nejvyšší výkon - nasazováno pro kritické části kódu
    - snadný přímý přístup k HW

C#

- Moderní jazyk se syntaxí podobnou C/C++
- Vše je objekt
- Neexistují ukazatele
- Hodnotové a referenční typy
  - hodnotové - int, float, String, struct
  - referenční - odpovídají ukazatelům, ale nikde se o tom nehovoří
- Automatická správa hromady - garbage collector
  - není třeba se starat o dealokování paměti (neexistuje explicitní mechanismus)
  - nepoužité objekty (= nejsou na ně žádné reference) se odstraňují automaticky
  - nedeterministické chování - spouští se "když je čas" nebo když dochází paměť

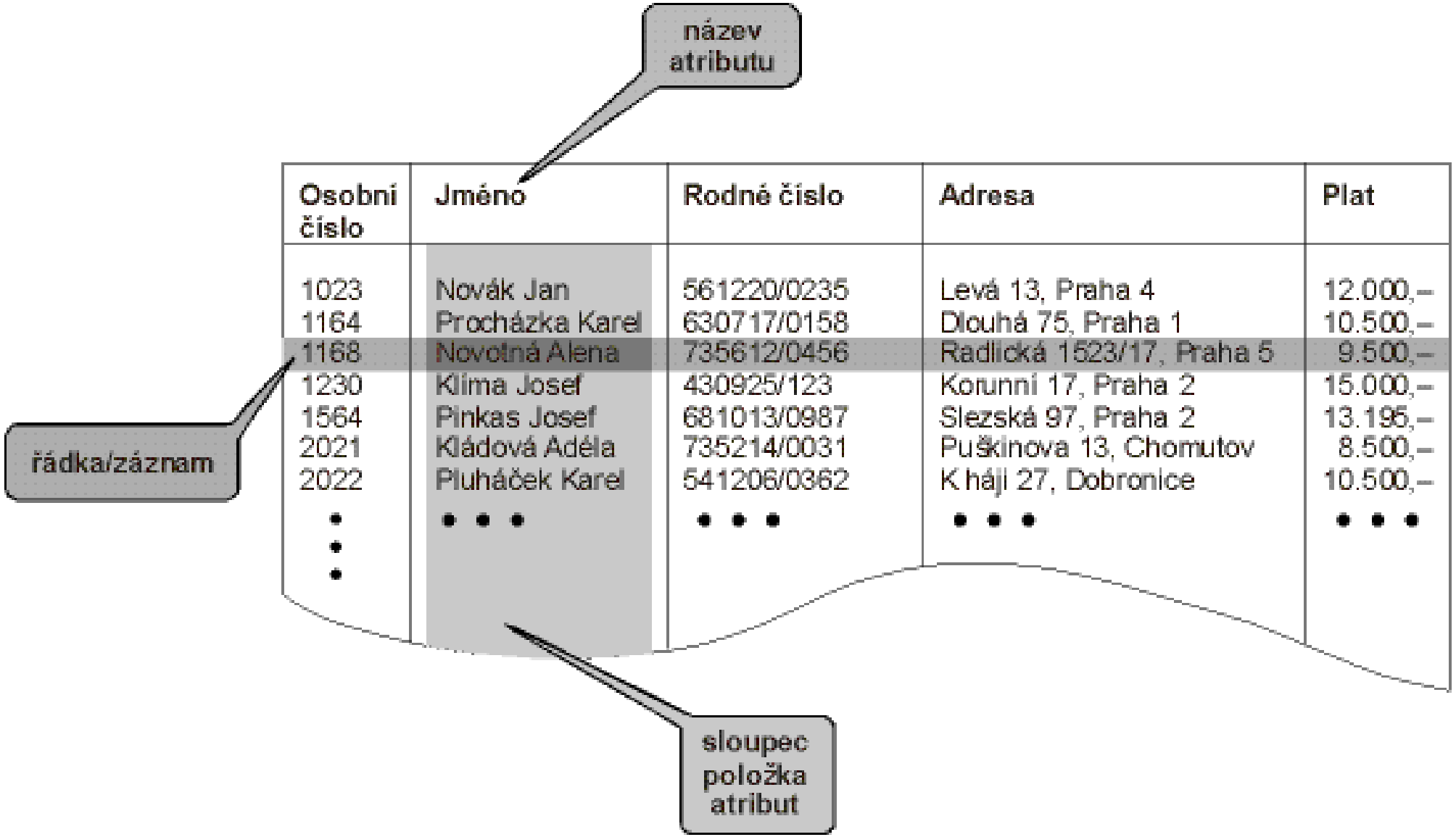
C# - II.

- Dědit lze pouze z jednoho předka
  - možno dědit více "abstraktních" předků = interface (rozhraní)
- Zrušeny hlavičkové soubory ve prospěch jmenných prostorů (`using`)
  - hierarchicky strukturovány
  - obsahují doplňující informace pro překladač i prostředí
  - pro další knihovny nutno do projektu přidat "reference"

Relační databáze – pojmy 1

- **Relační DB model** sdružuje data do **relací** (tabulek) obsahujících **n-tice** (řádky)
- **Tabulka** – sada záznamů s položkami (sloupci, atributy)
- **Sloupce** mají definován název, typ, příp. rozsah (doménu)
- **Index** – urychluje vyhledávání dat v tabulkách, pomalejší vkládání

Relační databáze – pojmy 1a



Osobní číslo	Jméno	Rodné číslo	Adresa	Plat
1023	Novák Jan	561220/0235	Levá 13, Praha 4	12.000,–
1164	Procházka Karel	630717/0158	Dlouhá 75, Praha 1	10.500,–
1168	Novotná Alena	735612/0456	Radlická 1523/17, Praha 5	9.500,–
1230	Klíma Josef	430925/123	Korunní 17, Praha 2	15.000,–
1564	Pinkas Josef	681013/0987	Slezská 97, Praha 2	13.195,–
2021	Kládová Adéla	735214/0031	Puškinova 13, Chomutov	8.500,–
2022	Pluháček Karel	541206/0362	K háji 27, Dobronice	10.500,–
•	• • •	• • •	• • •	• • •
•				
•				

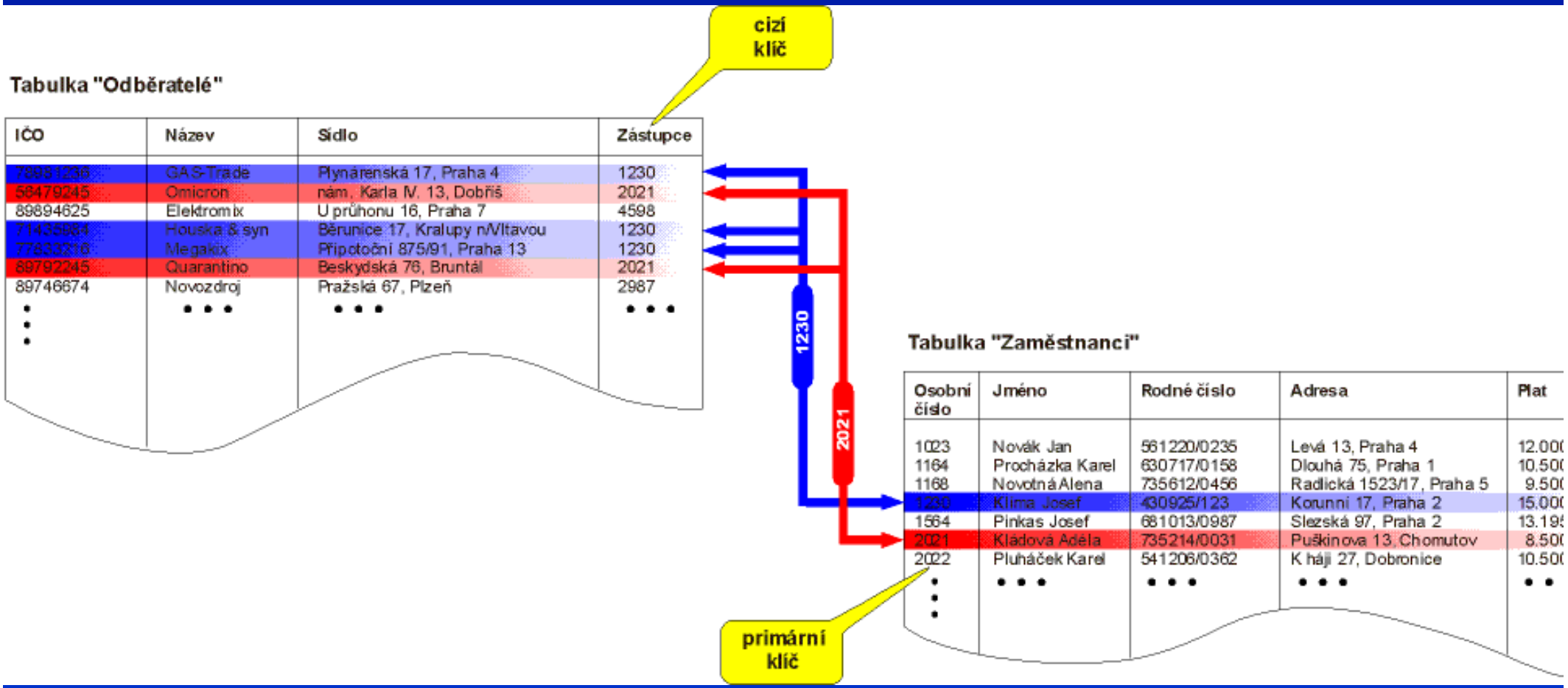
Relace mezi tabulkami

- Žádný vztah
- **1:1** = jeden záznam z tabulky odpovídá jednomu z druhé tabulky
- **1:N** = jednomu záznamu odpovídá více záznamů z jiné tabulky
- **M:N** = jednomu záznamu odpovídá více z druhé tabulky a naopak
  - velmi často odpovídá reálným datům
  - v DB nahrazováno typicky dvojicí tabulek 1:N

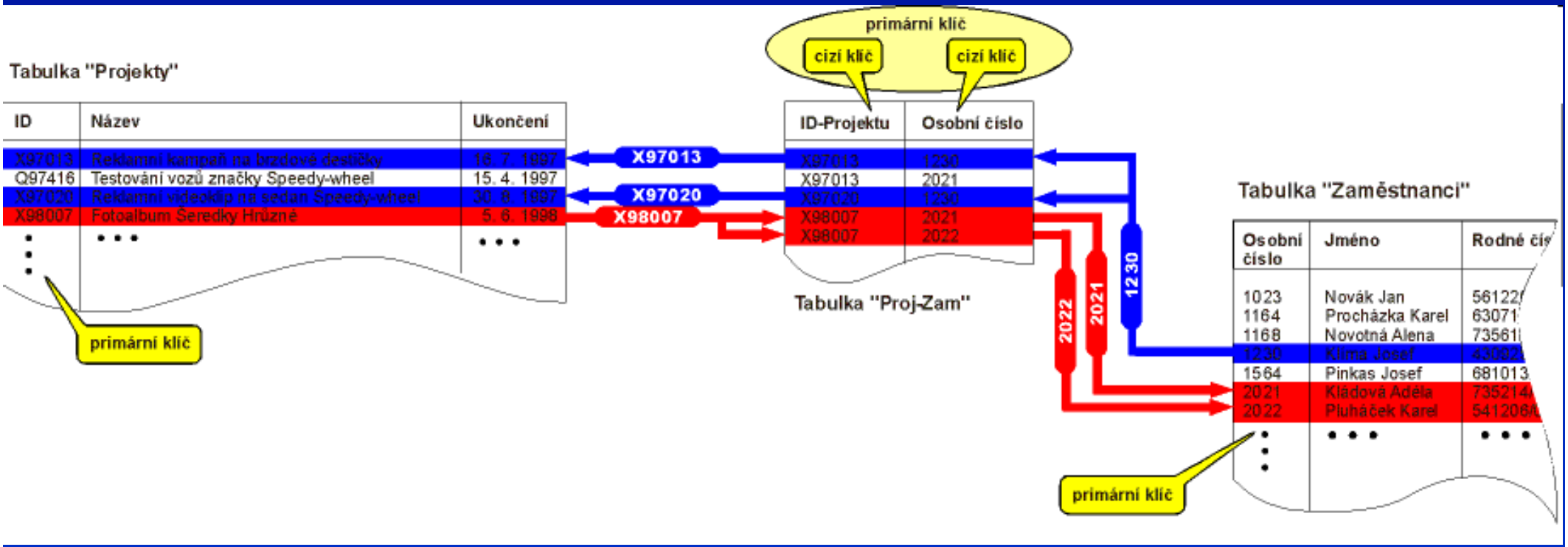
Relační databáze – pojmy 2

- **Primární klíč** – sloupec (nebo kombinace) s jedinečnou hodnotou pro každý řádek (záznam), neměnný po vytvoření
- **Referenční integrita, cizí klíče** – ve dvojici tabulek je jedna podřízená (slave) – kontroluje se, zda v nadřízené tabulce (master) je záznam s odpovídající hodnotou (logicky zřejmě primárního klíče)
  - Databáze by měla zajistit definované chování při smazání záznamu, který obsahuje cizí klíč z jiné tabulky
    - Nejčastěji blokuje smazání
- V db jazycích:
  - PRIMARY KEY
  - FOREIGN KEY

Relace 1:N – klíče pro vazbu



Relace M:N


primární klíč
cizí klíč
cizí klíč
Tabulka "Projekty"
ID
Název
Ukončení
X97013
Reklamní kampaň na brzdové destičky
16. 7. 1997
Q97416
Testování vozů značky Speedy-wheel
15. 4. 1997
X97020
Reklamní videoklip na sedan Speedy-wheel
30. 8. 1997
X98007
Fotoalbum Šeredky Hrůzné
5. 6. 1998
primární klíč
X97013
X97020
X98007
ID-Projektu
Osobní číslo
X97013
1230
X97013
2021
X97020
1230
X98007
2021
X98007
2022
Tabulka "Proj-Zam"
2022
2021
1230
Tabulka "Zaměstnanci"
Osobní číslo
Jméno
Rodné čís
1023
Novák Jan
1164
Procházka Karel
1168
Novotná Alena
1230
Klíma Josef
1564
Pinkas Josef
681013
2021
Kládová Adéla
2022
Pluháček Karel
primární klíč

SQL - Structured Query Language

- **DML – Data Manipulation Language** – příkazy pro manipulaci s daty:
  - SELECT, INSERT, UPDATE, DELETE
- **DDL – Data Definition Language** – příkazy pro definici dat:
  - CREATE, ALTER, DROP
- **DCL – Data Control Language** – příkazy pro správu transakcí a řízení přístupových práv
  - BEGIN, COMMIT, ROLLBACK
  - GRANT, REVOKE
- Ostatní příkazy ...

SQL – příkaz SELECT

```
SELECT
      [ALL | DISTINCT]
      {
        [jméno_tabulky. | alias_tabulky. | jméno_pohledu. ]
        {
          * | jméno_sloupce | jméno_sloupce AS alias
        }
      }[…, n]
     [INTO jméno_nové_tabulky]
     FROM
       {
         jméno_tabulky [AS alias_tabulky]
       }[…, n]
     [
       INNER JOIN | LEFT [OUTER] JOIN | RIGHT [OUTER] JOIN | CROSS JOIN | FULL
    OUTER JOIN
          jméno_tabulky [AS alias_tabulky]
          ON (podmínka)
     ][…, n]
     [ WHERE (podmínky) ]
     [ ORDER BY {výraz [ASC | DESC]}[…, n]  ]
```

SQL – příkaz SELECT – II

- Seznam sloupců výpisu doporučeno uvádět (ne jen * pro všechny)
  - Definuje se jejich pořadí
  - Přenáší se jen potřebné
- Pro více tabulek najednou se uvádí `tabulka.sloupec`
- Výběr z více tabulek
  - Vazba ve `WHERE` sekci – míchání s výběrovými podmínkami
  - `JOIN` – pomocí `ON (tbl1.sloupec1 = tbl2.sloupec2)` nebo `USING (id_sloupec)`
    - **`LEFT JOIN`** – všechny řádky z "levé" tabulky, nemusí mít vazbu do "pravé", chybějící hodnoty doplněny **`NULL`**
    - **`RIGHT JOIN`** – všechny řádky z "pravé"
    - **`FULL JOIN`** – vyskytující se alespoň v jedné tabulce
    - **`JOIN`** nebo **`INNER JOIN`** – existují záznamy v obou tabulkách

SQL – příkazy s daty

```
INSERT INTO tabulka (sloupec1,
  [sloupec2, ... ]) VALUES (hodnota1,
  [hodnota2, ...])
  INSERT INTO tabulka (...) SELECT (...)

UPDATE tabulka SET nazev_sloupecku =
  hodnota [, nazev_sloupecku = hodnota
  ...]
  [WHERE podminka = hodnota podm];

DELETE FROM [tab_name] WHERE
  [condition]
```

Databázové transakce

- Databázová transakce je skupina příkazů, které převedou databázi z jednoho konzistentního stavu do druhého
  - **A** - Atomic : Transakce se provede celá, nebo se neprovede vůbec.
  - **C** - Consistent : Po provedení transakce není porušeno žádné omezení.
  - **I** - Isolated : Operace uvnitř transakce jsou skryty před vnějšími operacemi.
  - **D** - Durale : Po ukončení transakce jsou data trvale uložena.
- Hranice skupin příkazů
  - `BEGIN` - začátek transakce
  - `COMMIT` - ukončení transakce a uložení dosažených výsledků do databáze
  - `ROLLBACK` - odvolání změn - není-li definován savepoint (místo po které lze provedené změny vrátit zpět) tak návrat do stavu před započetím vykonávání transakce.

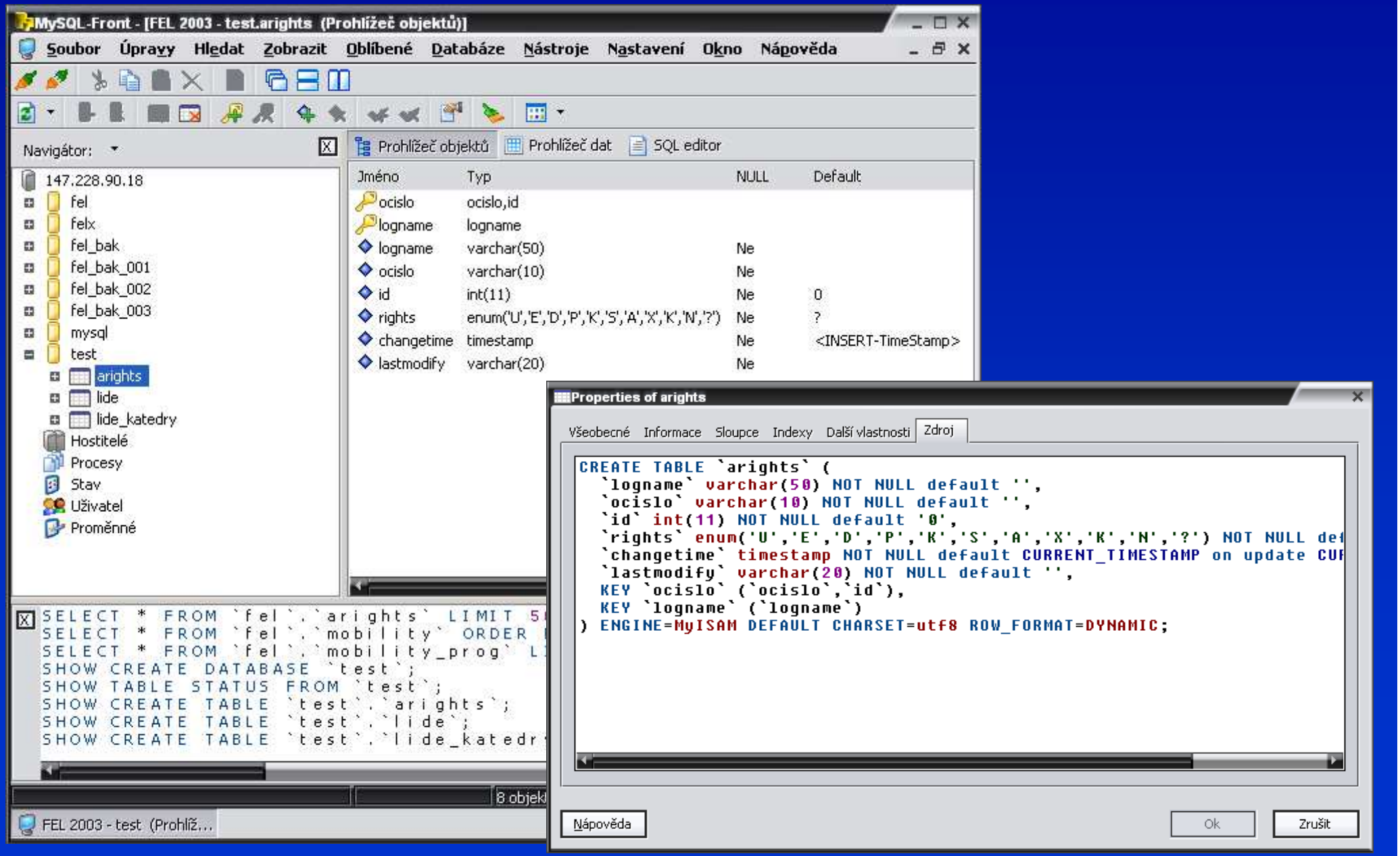
Normalizace databáze

- = úprava struktury tabulek tak, aby se odstranila redundatní data či optimalizovaly SQL dotazy
- Normální formy – NF:
  - 0. NF = tabulka obsahuje alespoň jeden atribut
  - 1. NF = každý atribut je nedělitelný (atomický)
  - 2. NF = každý atribut, který není součástí klíče je závislý na celé hodnotě klíče (pro složené klíče)
  - 3. NF = neklíčové atributy jsou vzájemně nezávislé
  - BCNF = varianta 3. NF i pro hodnoty klíče (všechna data)
  - 4. NF = primární klíč tvoří jen hodnoty, které mají skutečnou funkční souvislost
  - 5. NF = pro prim. klíče z 3+ atributů nesmí být párové cyklické závislosti = přidáním dalšího sloupce se rozpadne na více tabulek
- Vyšší NF splňuje všechny nižší
- Běžně si vystačíme 3. NF

Databázové stroje

- Služba/démon – server přes TCP/IP, pipe apod.
  - MySQL
    - po koupi Oraclem existuje komunitní fork – MariaDB
  - Microsoft SQL Server 2000, 2005 nebo 2008 (R2), 2012
  - PostgreSQL
  - SyBase -> nyní FireBird
  - Oracle
- Embedded databáze – připojený soubor
  - SQLite
  - FireBird
  - SQL Server 2005/8 Compact Edition (= Everywhere)
  - XML
  - Access .MDB – od WinXP součástí systému příslušný MDAC ovladač, Access (Office) je jen frontend

Správa databází



Programátorský přístup

- Přímé volání funkcí DB interface (typicky C++)
- Pohodlnější objektový přístup – ADO.NET
  - SqlConnection
    - ConnectionString
  - SqlDataAdapter
    - naplnění DataSet (více DataTable s příp. relacemi)
  - SqlDataReader
    - sekvenční procházení
  - SqlCommand
    - možnost parametrizování

Objektově relační mapování - ORM

- Objekty reálného světa představovány entitami
  - v DB řádek (množina řádků) tabulky
  - v programu objekty (instance třídy)
- ORM nástroje mapují DB data na třídy/objekty v programu
  - není třeba pracovat přímo se SQL dotazy
  - příslušná konverze typů apod. je automatická
  - nezávislost na konkrétní DB platformě
  - možná cesta k dědičnosti
- Persistence
  - provozní data v paměti musí být aktuální vzhledem k databázi
  - v případě ukončení aplikace musí být zajištěna aktualizace DB – vše podle pravidel ACID (= typicky transakce)

Příklady kódu

```
_conn.Open();
using (FbCommand cmd = _conn.CreateCommand())
{
  cmd.CommandText =
    "INSERT INTO provoz (id_typ_akce, podrobnosti, udalost) VALUES (@typ, @podrob, @cas)";
  cmd.Parameters.Add("@typ", typAkce);
  cmd.Parameters.Add("@podrob", strDetail);
  cmd.Parameters.Add("@cas", datumCas);

  iRows = cmd.ExecuteNonQuery();
}
_conn.Close();
```

```
_conn.Open();
using(MySqlCommand cmd = _conn.CreateCommand())
{
  cmd.Parameters.Add("ocislo", ocislo);
  cmd.Parameters.Add("id", idPredmet);
...
  cmd.CommandText = "SELECT COUNT(*) FROM lide_predm WHERE ocislo = ?ocislo AND id_predmet = ?id";

  Object o = cmd.ExecuteScalar();
  if ((o == null) || (o == DBNull.Value) || (Convert.ToInt32(o) < 1))
    cmd.CommandText = @"INSERT INTO lide_predm (ocislo, id_predmet, ...) VALUES (?ocislo, ?id, ...)";
  else
    cmd.CommandText = @"UPDATE lide_predm SET garant = ?garant WHERE ocislo = ?ocislo AND id_predmet = ?id";

  bbResult = cmd.ExecuteNonQuery() > 0;
}
_conn.Close();
```

Informační zdroje - I

- http://www.microsoft.com/visualstudio/en-us/
- http://www.microsoft.com/cze/msdn/vstudio/2010/
- http://csharp.aspone.cz/default.aspx?AspxAutoDetectCookieSupport=1
- http://projektysipvz.gytool.cz/ProjektySIPVZ/Default.aspx?uid=1
- http://poznavame-c-msnet.wz.cz/ - offline verze rozsáhlého seriálu na zive.cz
- http://www.zive.cz/clanky/tri-veterani-uzivatelske-srovnani-integrovanych-prostredi-pro-jazyk-c/sc-3-a-135680/default.aspx
- http://monotouch.net/
- http://www.mono-project.com/Main_Page
- http://www.go-mono.com/monodroid/ - resp. http://monodroid.net/
- http://android.xamarin.com/

- ... použijte google ☺

Informační zdroje - II

- http://interval.cz/serialy/sql-structured-query-language/
- http://www.dbsvet.cz/index.php
- http://cs.wikipedia.org/wiki/SQL
- http://www.manualy.net/article.php?articleID=13
- http://www.mysql.com
- http://www.firebirdsql.org
- http://www.sqlite.org
- http://msdn.microsoft.com/vstudio/express/sql/
- http://www.kosek.cz/clanky/iweb/12.html
- http://cs.wikipedia.org/wiki/Objektov%C4%9B_rela%C4%8Dn%C3%AD_mapov%C3%A1n%C3%AD
- http://programujte.com/clanek/2008071900-normalizace-relacnich-databazi/
- http://www.heidisql.com/
- https://cs.wikipedia.org/wiki/MariaDB

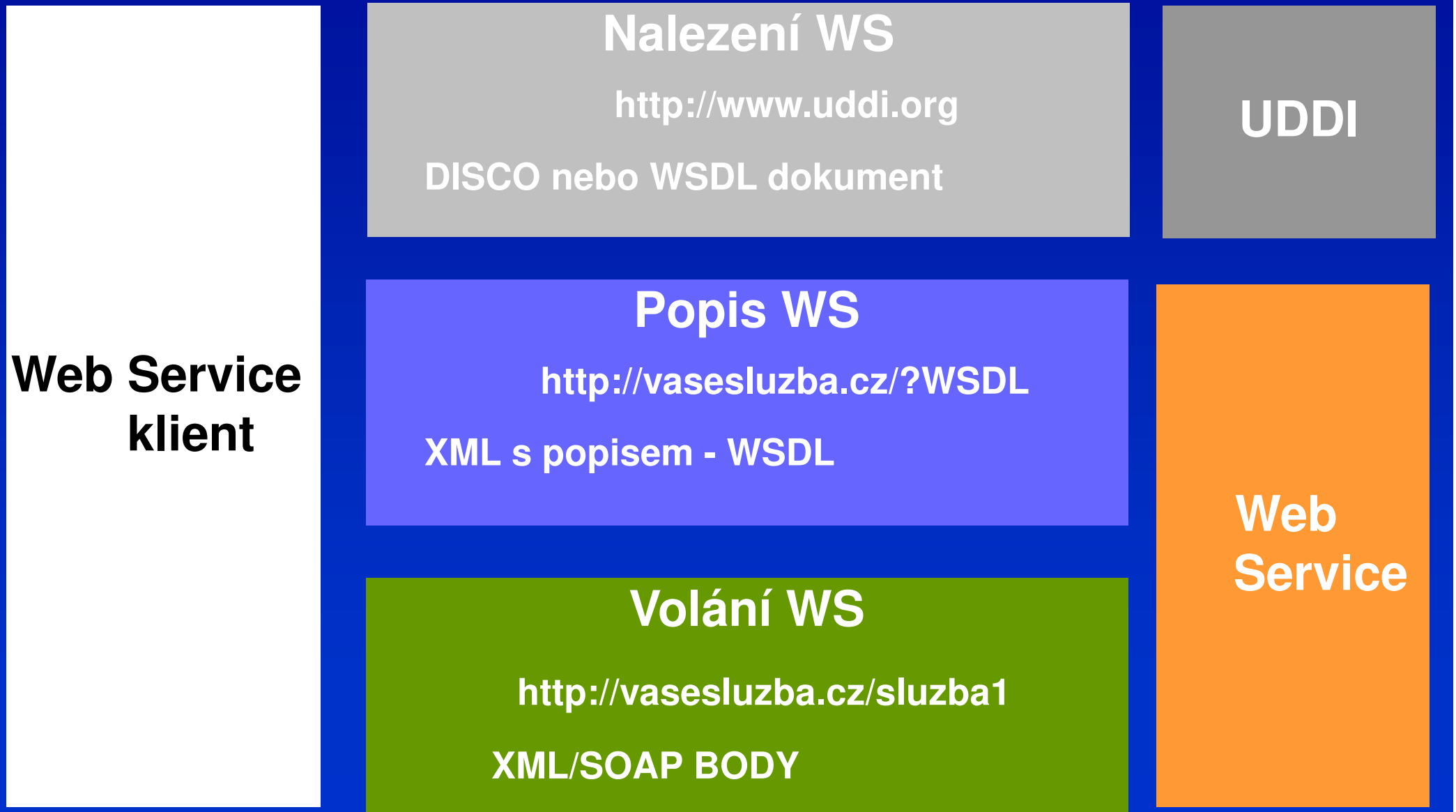
2. přednáška

- WebService – co to je
- XML – značkovací jazyk
- HTTP protokol
- Realizace 1 – PHP
- Realizace 2 – ASP.NET
- ASP.NET a tenký klient

WebServices

- **WS** = "Webové služby"
- komunikace M2M založená na otevřených standardech
  - **HTTP** (HyperText Transfer Protocol)
  - **XML** (eXtensible Markup Language)
  - **SOAP** (Simple Object Access Protocol)
  - **UDDI** (Universal Description, Discovery, and Integration)
  - **WSDL** (Web Services Description Language)
  - **REST** (REpresentative State Transfer)
  - **JSON** (JavaScript Object Notation)
- **SOA** = Service Oriented Architecture
- *BuzzWord* let 2004-5

WebServices - architektura



Značkovací jazyky -> XML

- ML – markup languages
- HTML – HyperText ML
  - volnější díky toleranci prohlížečů
- XML - eXtensible ML (+DTD apod.)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD
    HTML 3.2//EN">
<HTML>
  <HEAD>
    <TITLE>HTML dokument</TITLE>
  </HEAD>
  <BODY>
    <H1>Nadpis</H1>
    <P>První odstavec
  </BODY>
</HTML>
```

```
<firma>
  <název>Botička a Verpánek, s.r.o.</název>
  <sídlo>
    <ulice>Přípotoční 17</ulice>
    <město>Praha 4</město>
    <psč>140 00</psč>
    <telefon>02/90123456</telefon>
  </sídlo>
  <oborČinnosti>
    Oprava obuvi, koženého zboží, výměna zipů v
    kabelkách a taškách, výroba zdravotnické a
    ortopedické obuvi na zakázku.
  </oborČinnosti>
</firma>
```

HTTP protokol

- Jednoduchá implementace metody "požadavek–odpověď" v prostředí "klient-server"
- Typicky požadavek WWW stránky dané URL adresou, odpověď je text stránky v HTML formátu
- Případně je možno přenášet binární data s příslušnou hlavičkou (obrázky, PDF, ...)
- Bezstavový – mezi požadavky si sám server nic „nepamatuje“

Struktura SOAP

Envelope

Header

Body

... Fault

```
<soap:Envelope>
  <soap:Header>
    <s:credentials xmlns:s="urn:examples">
      <username>dave</username>
      <password>evad</password>
    </s:credentials>
  </soap:Header>
  <soap:Body>
    <CreateExpenseReportResponse>
      <CreateExpenseReportResult>
        <EmailName>rhoward</EmailName>
        <Expenses>
          <Cost>priceless</Cost>
        </Expenses>
      </CreateExpenseReportResult>
    </CreateExpenseReportResponse>
  </soap:Body>
</soap:Envelope>
```

Princip WS typu REST

- Zjednodušení komunikace na minimum
- Požadavek zakódován v URL
- Odpověď soubor XML
  - Typicky minimální hlavička
  - Strukturovaná „čistá data“

```
http://muj.server.cz/sluzba.php?param1=neco&param2=0
...
<moje_data>
  <item>
    <data>...</data>
  </item>
  <item>
    ...
  </item>
</moje_data>
```

Předávání dat ve formátu JSON

- Obsahuje pouze data, typicky ve formátu "inicializace pole" v JS
- Typicky kombinace klíč-hodnota
- Žádné formátovací ani strukturální informace = minimální režie
- Zpracování dat
  - JS – buď rovnou pomocí funkce eval() nebo pomocí knihoven
  - Ostatní – typicky funkce z knihoven

```
[
  {"name": "Cerna sanitka", "tvname": "CT1"},
  {"name": "Comeback", "tvname": "Nova"},
  {"name": "Pratele", "tvname": "Prima"}
]
```

- Zdroje informací:
  - http://www.json.org/json-cz.html
  - https://www.zdrojak.cz/clanky/json-jednotny-format-pro-vymenu-dat/

WebService a PHP

- Pro REST přímo podpora XML v PHP 5+
- NuSOAP - Web Services Toolkit for PHP
  - http://sourceforge.net/projects/nusoap/
  - ověřený provoz
    - funguje volání **xxx.php?wsdl**
    - problémy s kódováním (defaultně ISO-8859-1)

```
$server->wsdl->addComplexType(          // NuSOAP pridani noveho typu = struktury
    'SOAPStruct...',                    // nazev struktury pro prenos dat
    'complexType',
    'struct',
    'all',
    '',
    array(                              // jednotlive prvky definovany v poli
          'iId' => array('name'=>'iId','type'=>'xsd:int'),
                    ...
    )
);
$server->register('GetData',            // pridat novou funkci v ramci WS
    array('strZkratka'=>'xsd:string'),  // jeden parametr typu retezec
    array('return'=>'tns:SOAPStruct...'), // navratova hodnota "moje" struktura
    $ns
);
```

Příklad WS PHP < v5

```
include_once("/include/nusoap/nusoap.php");          // podle mista include
$server = new soap_server();

$ns = 'http://my_server.cz/';                        // doplnit server
$server->configureWSDL('FELnuDB', $ns);
$server->wsdl->schemaTargetNamespace = $ns;

$server->register("EchoString",                     // registrace nove funkce WS
    array('input' => 'xsd:string'),                 // parametr "string"
    array('return' => 'xsd:string'),                // výsledek "string"
    $ns
);
function EchoString($input)                         // implementace funkce
{
    if (is_string($input))
        return new soapval('return', 'string', "Input String: " . $input);
    else
        return new soap_fault("Client", "", "Param musi byt string.");
}
$server->service($HTTP_RAW_POST_DATA);        // povinne ukonceni PHP skriptu
```

XML (WebService) a PHP5+

- Součástí jazyka přímo DOMDocument a DOMXPath
- Vyžaduje libxml PHP extension (--enable-libxml)

```
class ConfigClass
{
  var $rok, $nazev, $mailInfo;
  var $_strXmlFileName = “konfigurace.xml”;
  var $_xml; var $_xpath;
  var $_aXPathRok; var $_aXPathLast;

  function LoadParams()
  {
    $this->_xml = new DOMDocument();
    $this->_xml->load($this->_strXmlFileName);
    $this->_xpath = new DOMXpath($this->_xml);

    $this->_aXPathRok = $a = $this->_xpath->query("//konference[last()]");

    if (is_null(!$this->_aXPathRok) || ($this->_aXPathRok->length != 1))
      trigger_error("Cannot find active data in configuration file !", E_USER_ERROR);

    $this->rok = $this->_aXPathRok->item(0)->attributes->getNamedItem("rok")->nodeValue;
    $this->nazev = $this->_xpath->query("./nazev[@lang = 'en']",
      $this->_aXPathRok->item(0))->item(0)->nodeValue;
    $this->mailInfo = $this->_xpath->query("./mail/info",
      $this->_aXPathRok->item(0))->item(0)->nodeValue;
...
```

ASP - Vzhled a kód lze oddělit

„klasické“ASP

1 soubor

kód

<html tagy>

Form1.aspx

ASP.NET, Visual Studio .NET

2 soubory

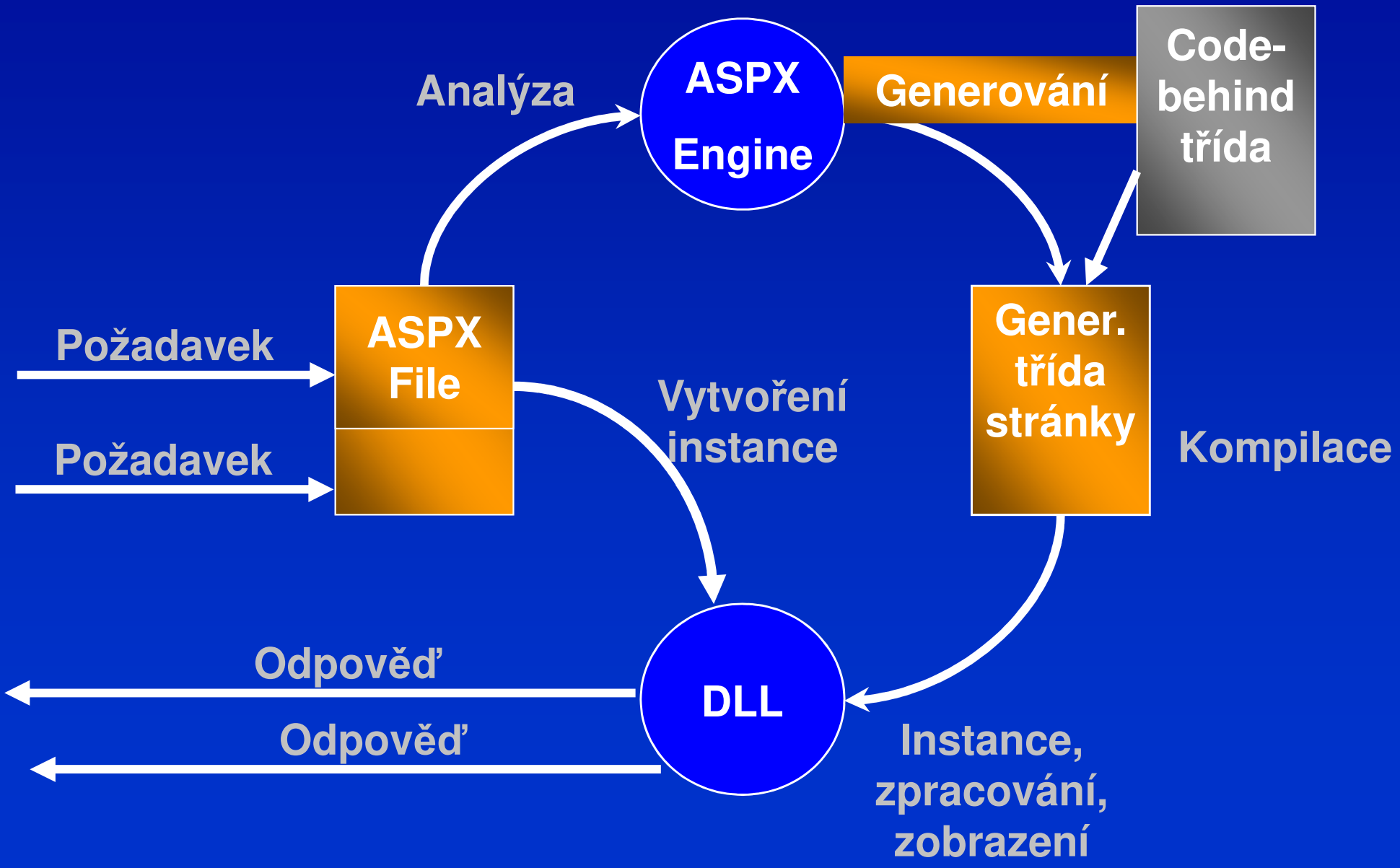
<html tagy>

Form1.aspx

kód

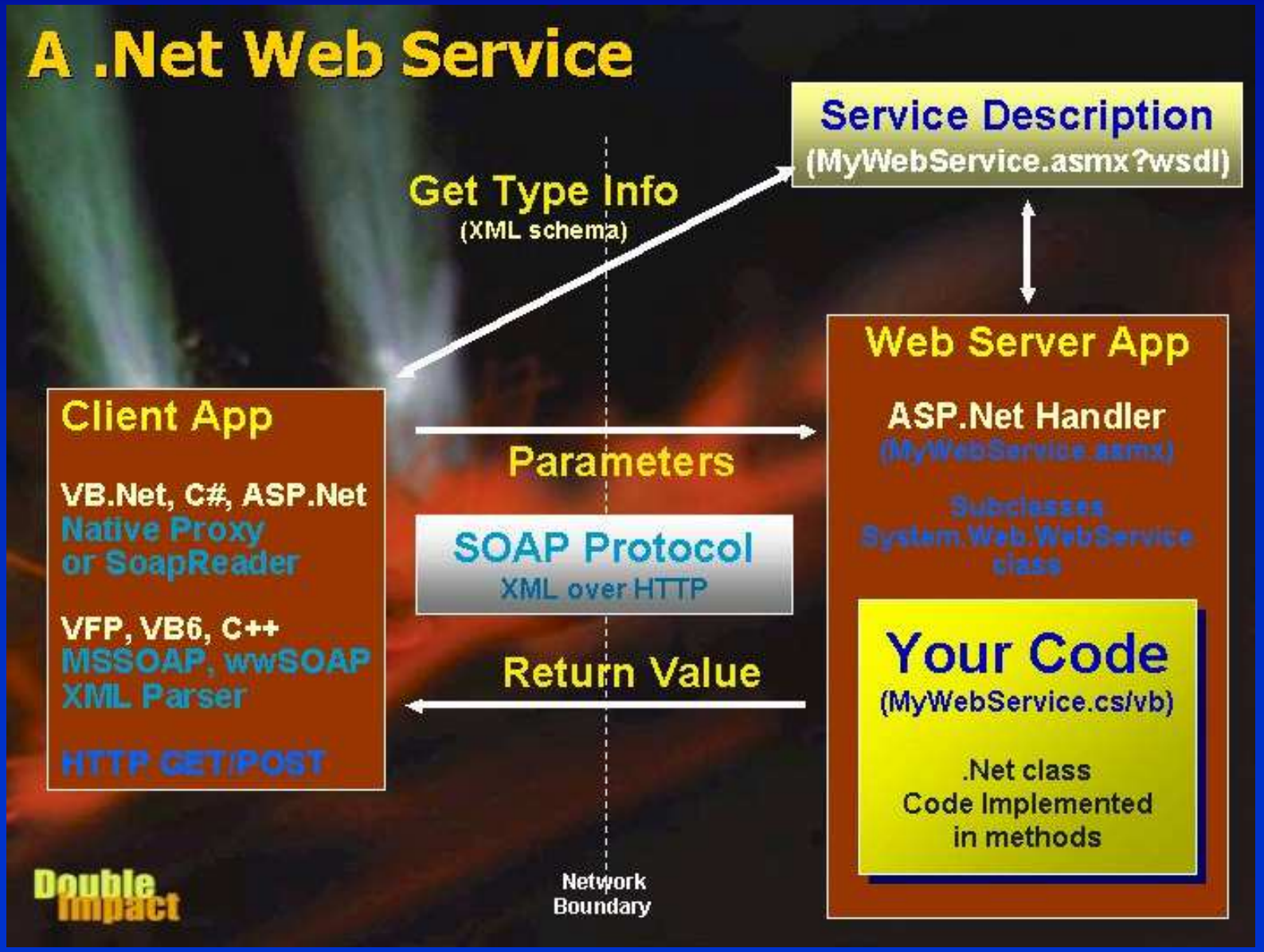
Form1.aspx.cs

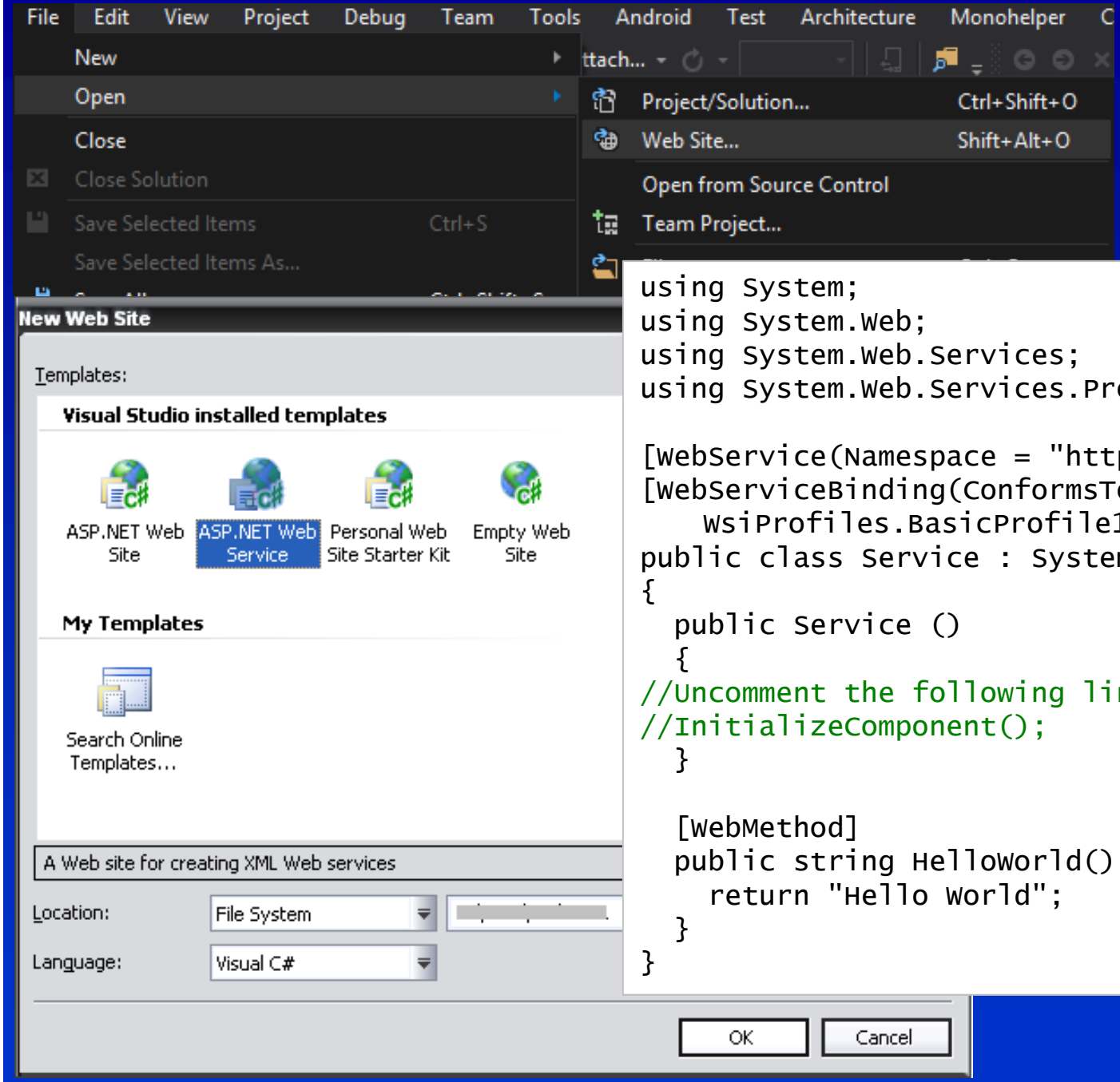
Kompilace ASP.NET

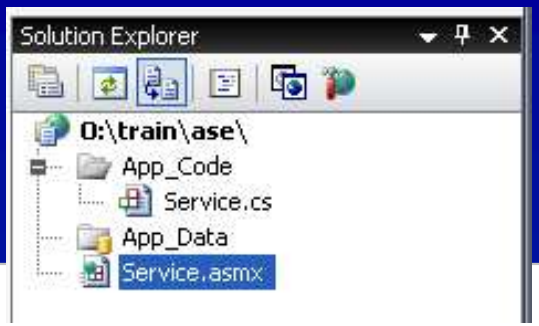


WS a ASP.NET

- http://www.west-wind.com/presentations/dotnetwebservices/DotNetWebServices.asp

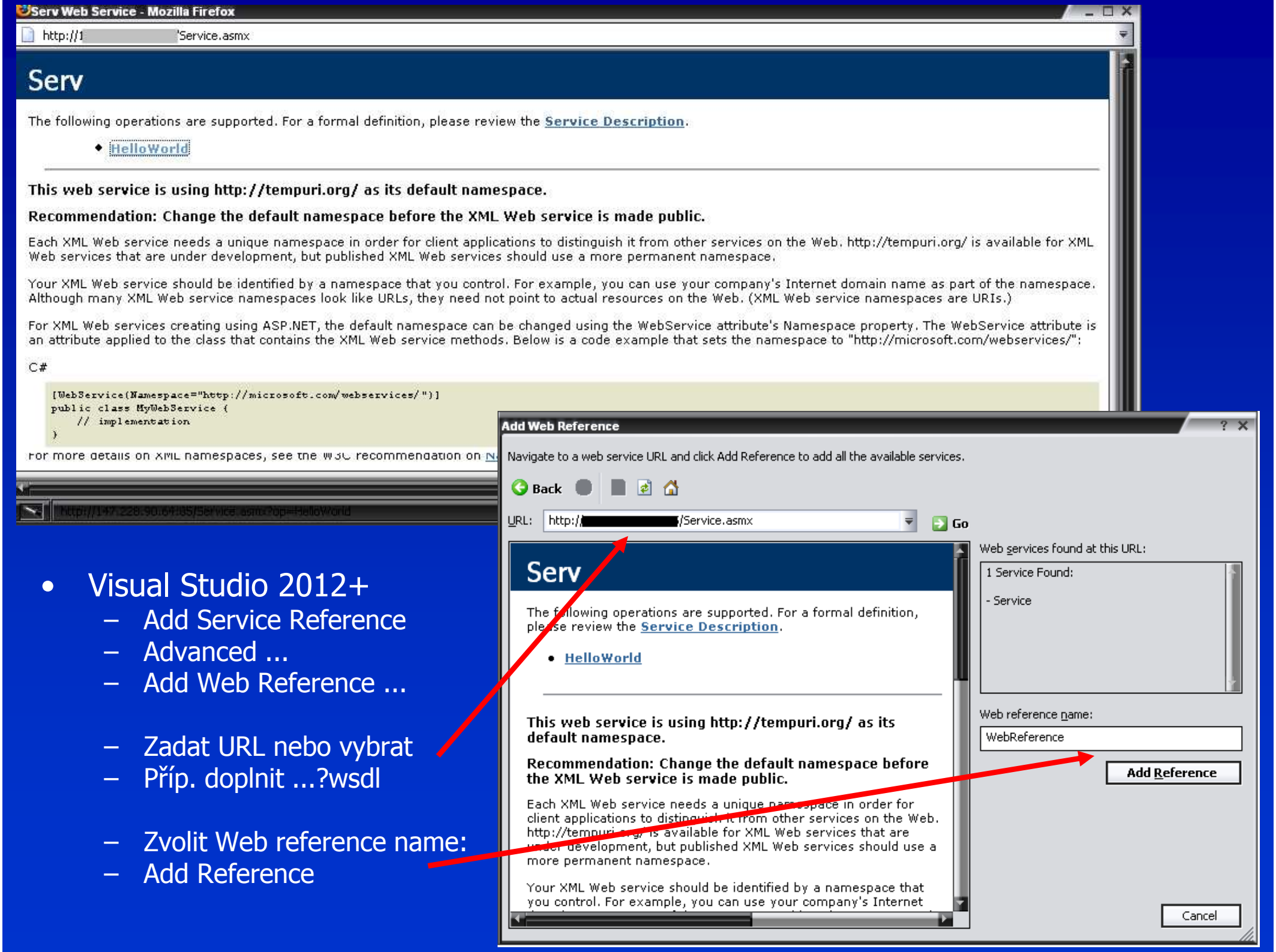




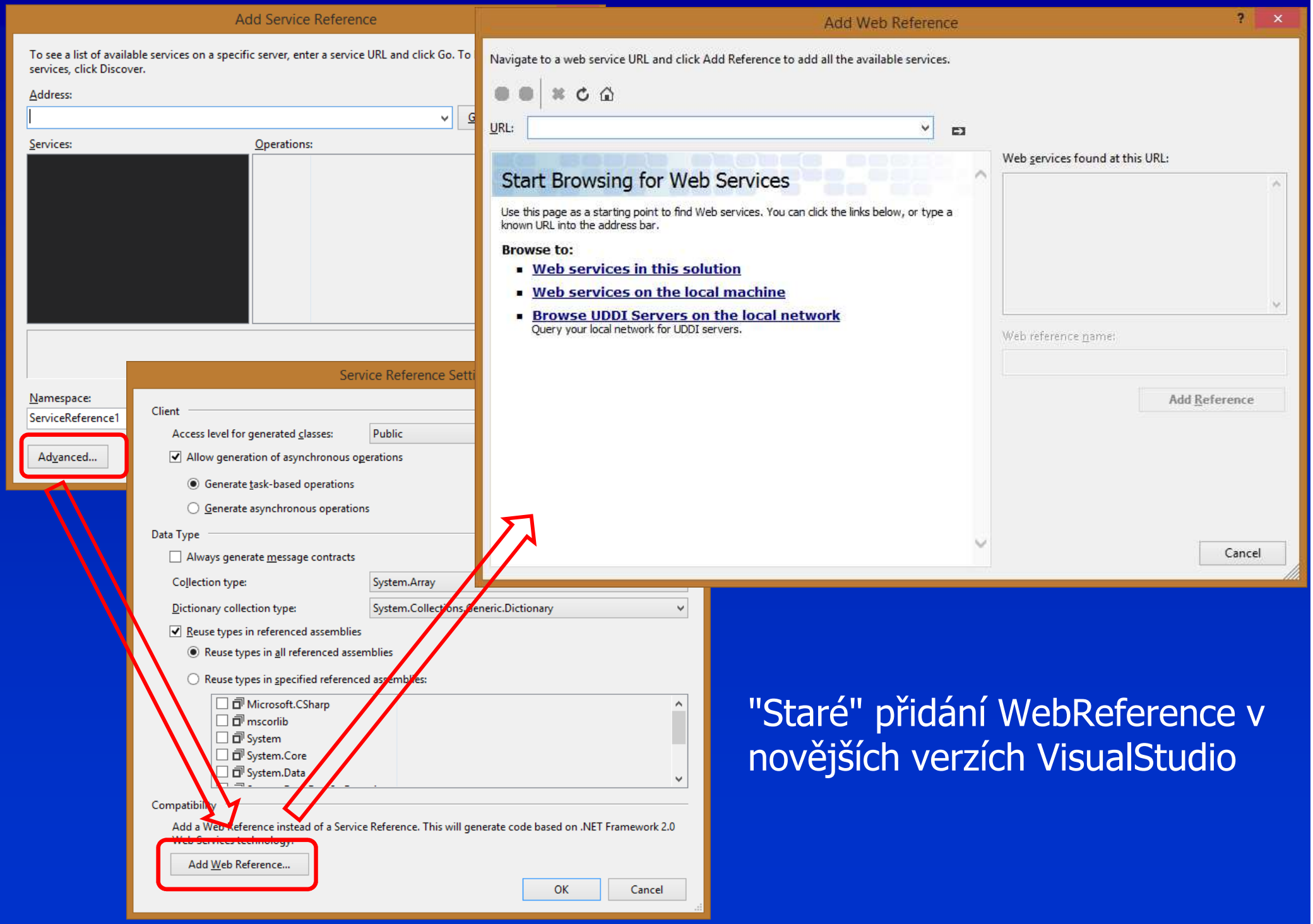



```
using System;
using System.Web;
using System.Web.Services;
using System.Web.Services.Protocols;

[WebService(Namespace = "http://tempuri.org/")]
[WebServiceBinding(ConformsTo =
    WsiProfiles.BasicProfile1_1)]
public class Service : System.Web.Services.WebService
{
  public Service ()
  {
//Uncomment the following line if using designed
//InitializeComponent();
  }

  [WebMethod]
  public string HelloWorld() {
    return "Hello World";
  }
}
```

```
<%@ WebService Language="C#" CodeBehind="~/App_Code/Service.cs" Class="Service" %>
```



- Visual Studio 2012+
  - Add Service Reference
  - Advanced ...
  - Add Web Reference ...
  - Zadat URL nebo vybrat
  - Příp. doplnit ...?wsdl
  - Zvolit Web reference name:
  - Add Reference



"Staré" přidání WebReference v novějších verzích VisualStudio

Příklady veřejných WS

- http://www.csas.cz/csws/changerates/ChangeRates.jws
  - Kurzovní lístek ČSAS
  - Vypnutý od 2016 ☹
- http://webservices.gama-system.com/exchangerates.asmx
  - Kurzovní lístek "nějakých" slovinských bank
  - Vrací XML data, požadavky viz. http://webservices.gama-system.com/descriptions-ER-slo.asp?strFunction=Common#Common
- http://www.webservicex.net/globalweather.asmx
  - Počasí celosvětově (momentálně nefunguje – od (?) 2017)
- http://openweathermap.org/
  - Velké možnosti formátů dat
  - Vyžaduje API key, free omezen na max. 60 requests/minute
    - https://openweathermap.org/price
- http://www.soapclient.com/soaptest.html
  - Testovací nástroj
- Alternativně plugin do prohlížeče Firefox
  - https://addons.mozilla.org/en-US/firefox/addon/soa-client/
- http://www.otevrenadata.cz/otevrena-data/zdroje-dat/
  - OpenData – různá data z ČR – nejen ministerstva
  - Např. http://doprava.plzensky-kraj.cz/site/page?view=od-traffic
    - Aplikace: http://opentransportnet.eu/web/guest/pilsen-traffic-volumes

3. přednáška
Komunikace po internetu

- ISO/OSI model a vrstvy
- IP, Sockety, UDP, TCP
- Ostatní protokoly
- FTP
- HTTP – GET a POST
- Využitelné třídy v .NET 2 (3.5+)

ISO/OSI model

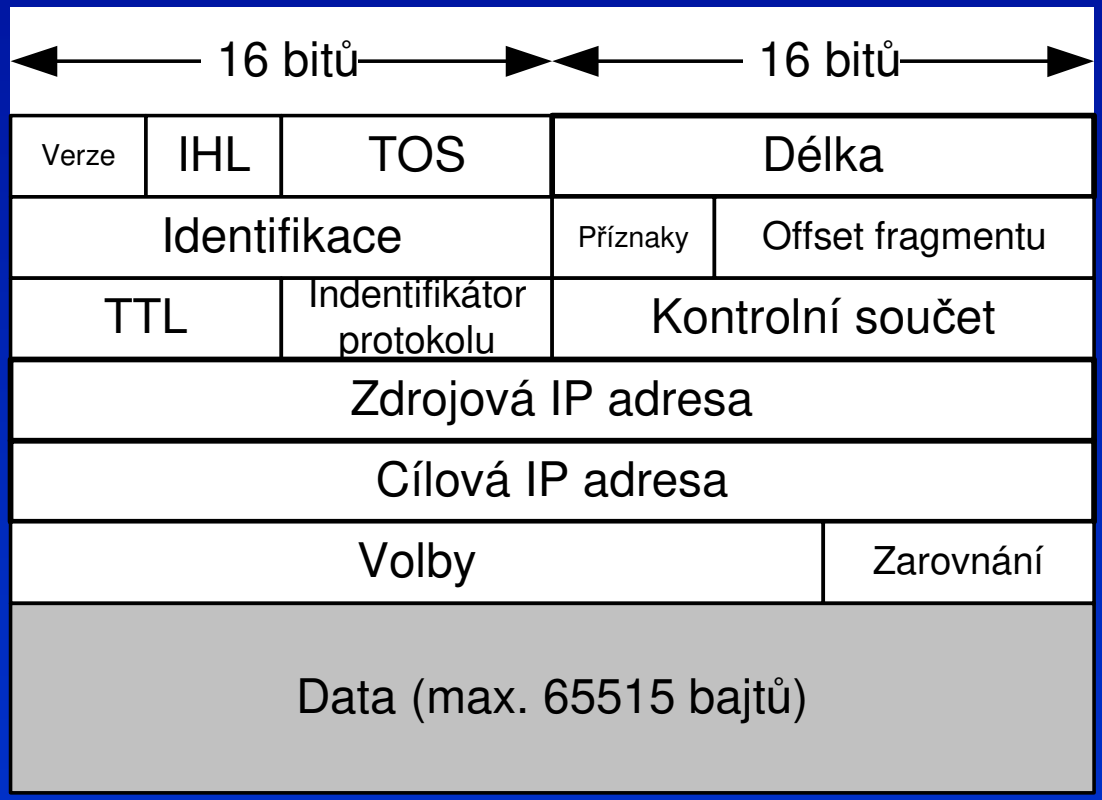
Aplikační	SMTP – Simple Mail Transport Protocol FTP – File Transfer Protocol
Prezentační	
Relační	
Transportní	TCP – Transmission Control Protocol UDP – User Datagram Protocol
Síťová	IP – Internet Protocol ICMP – Internet Control Message Protocol ARP – Address Resolution Protocol
Spojová	IEEE 802.3 Ethernet
Fyzická	

Protokol IP

- Datagram – „nespolehlivá“ data
- IP datagram – zapouzdřen do vyšších protokolů
- Často využíván „tečkový“ dekadický zápis
- IPv4 – adresy jako 32-bitová čísla
  - Adresové rozsahy vyčerpány v 2011
    - Používají se různé triky – např. NAT = překlad adres = "za" jednou IP adresou serveru/routeru se nachází více zařízení
- IPv6 – verze 6
  - Celkem 2^{128} adres = není potřeba NAT apod.
  - Zjednodušeno směrování a další servisní služby
  - Vyšší protokoly nedotčeny, nutno přizpůsobit "pouze" ICMP, DHCP apod.

IP datagram

- Hlavička rámce



- Minimální velikost IP datagramu 576 bajtů
  - 20 hlavička + 512 data + 44 doplňující volby nebo hlavička nižších vrstev

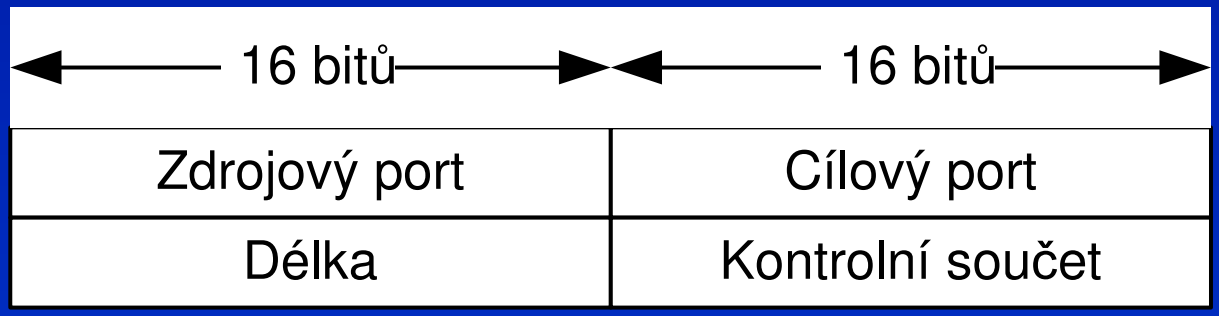
IP protokol – II

- Datagramy mohou cestovat „rozsekány" mezi více rámců
  - Identification určuje společné rámce
  - V příznacích nastaven bit MF „More Fragments"
  - Fragment Offset
- Kontrolní součet – 16-bitový jedničkový doplněk 16-bitového součtu hlavičky (pouze !)
- TTL (Time To Live) – ochrana proti zacyklení
  - Každý směrovač zmenší tuto hodnotu o jedničku
  - Pokud TTL == 0, datagram se zahodí (= vypršela životnost)

- Porty – využití více komunikačních kanálu na jednom stroji (resp. rozhraní = IP adrese)
  - Společně s IP adresou jednoznačně identifikují datový proud mezi aplikacemi
  - Pevně nastaveny (dohodnuty) pro známé aplikace

UDP protokol

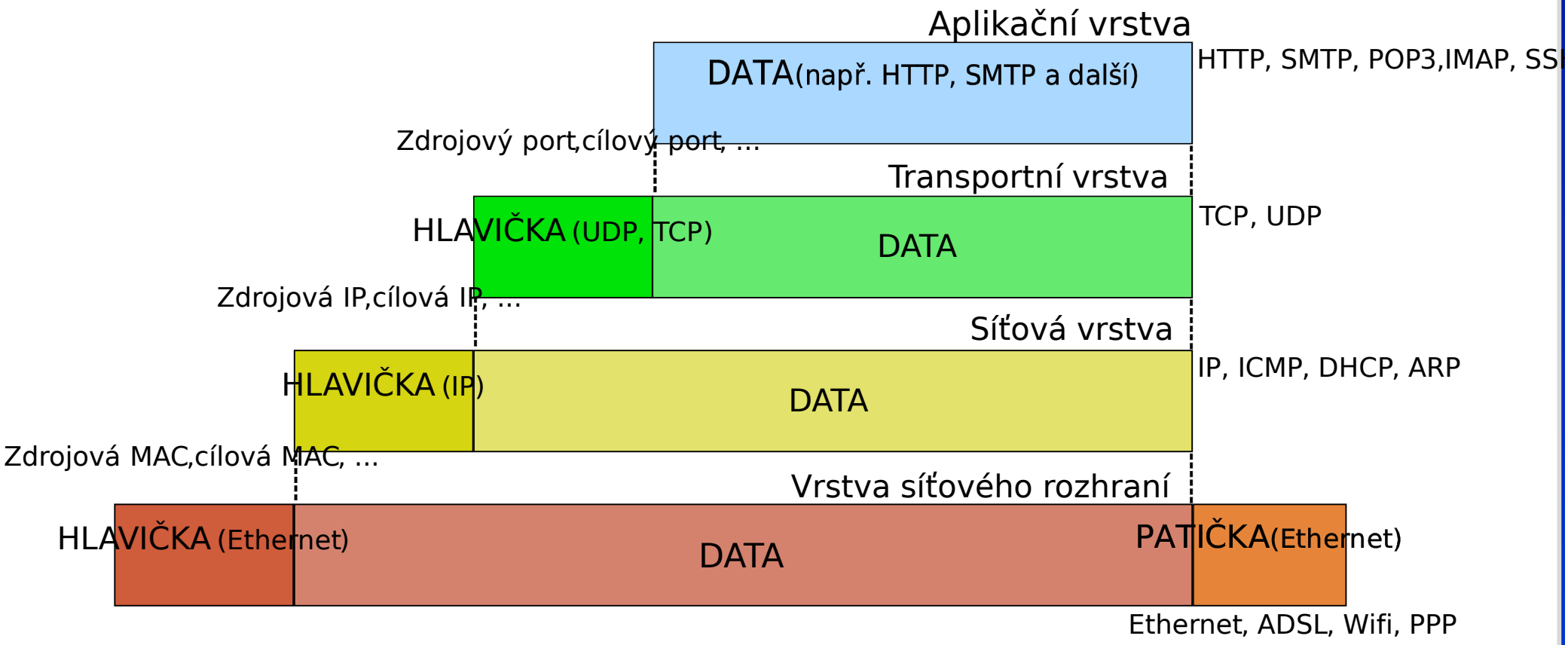
- Přidává k IP čísla portů
- Přidává kontrolní součet i přes data
- Hlavička UDP:



  - Délka včetně hlavičky
  - Kontrolní součet zahrnuje i IP adresu ve formě pseudohlavičky (RFC768)

- **Stejně jako IP nezaručuje dodání !!**

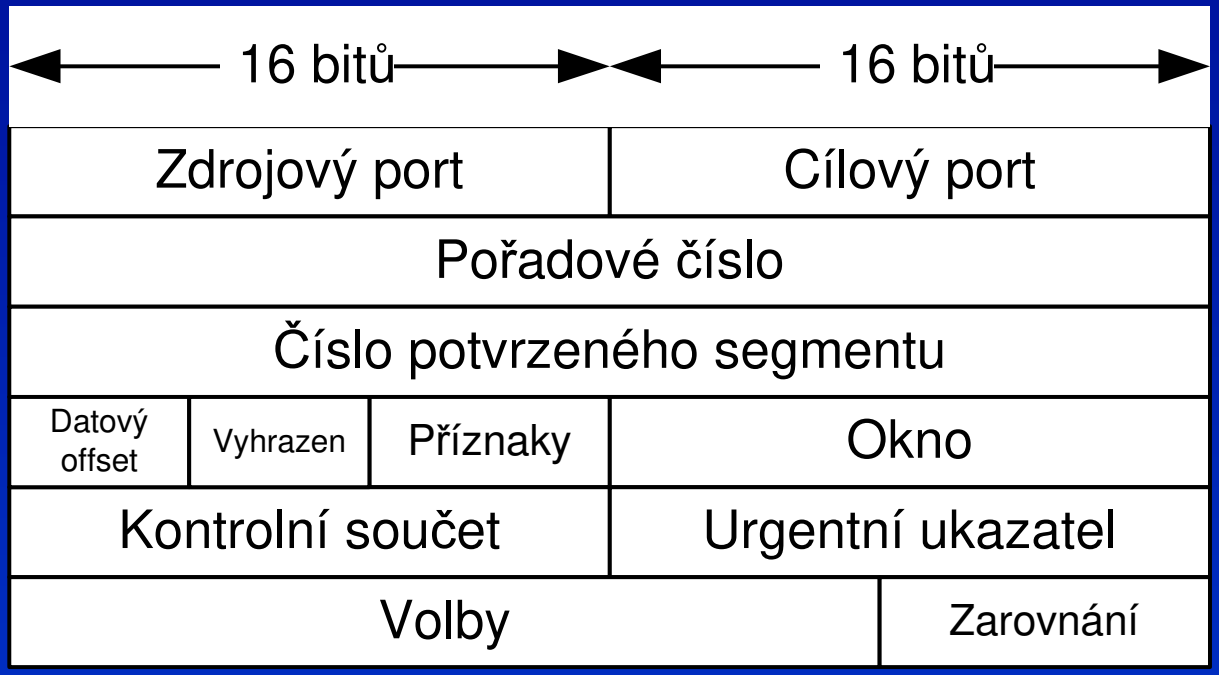
ZAPOUZDŘENÍ DAT V SÍTI TCP /IP



TCP protokol

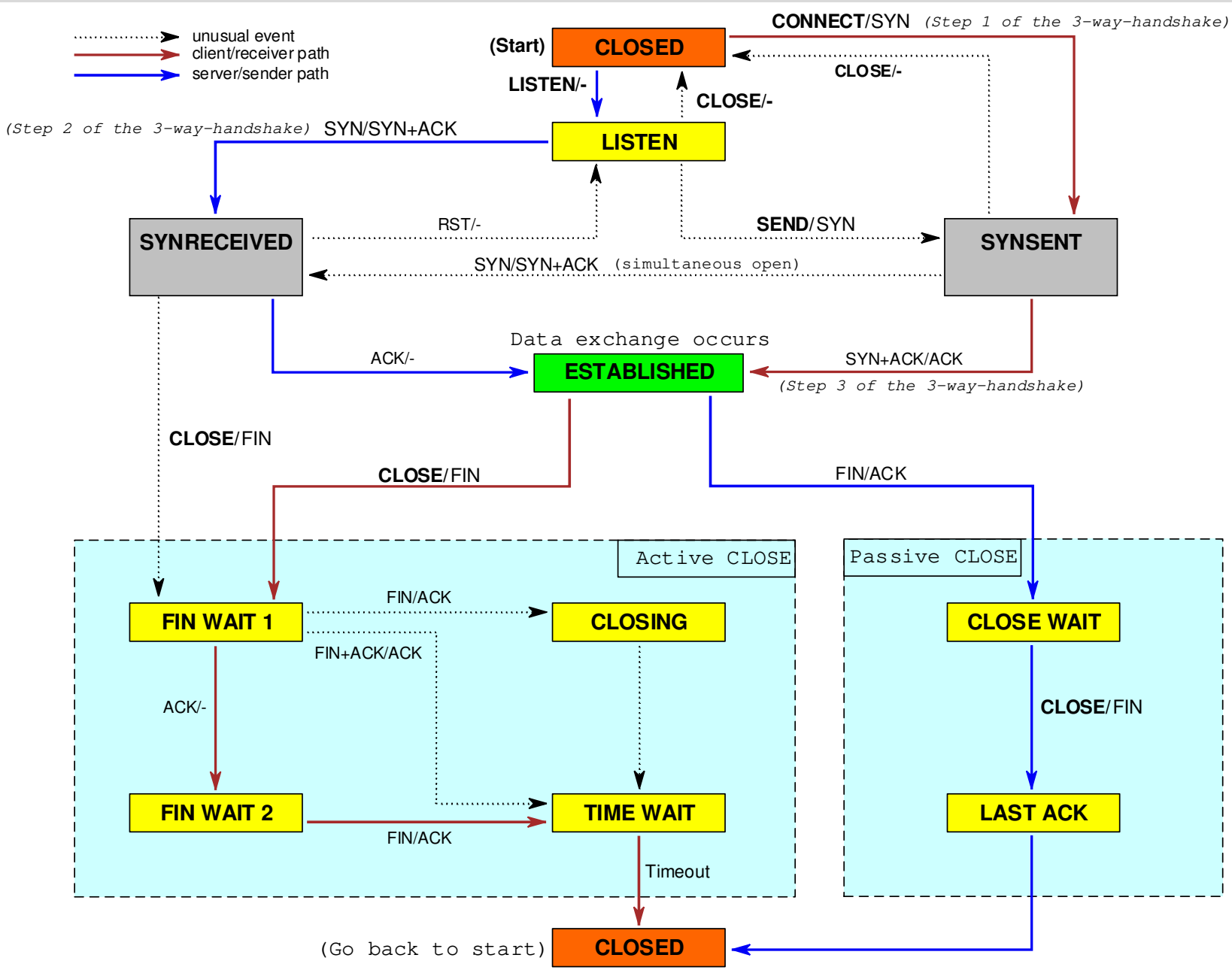
- Plně duplexní kanál mezi aplikacemi
- Řeší ztracené či zdvojené datagramy a složení v místě příjemce
- Má fáze "navázání spojení", "přenos dat" a "ukončení spojení"
- Data rozdělena do paketů tak, aby se vešla do IP rámce
- Využito plovoucí okno (sliding window)
  - nemusí se čekat na potvrzení každého paketu
  - Možno vysílat „napřed“

Hlavička TCP



- Počet bajtů se vypočte jako „zátěž" datagramu IP mínus velikost TCP hlavičky
- Číslo potvrzeného segmentu může být kumulativní – za více segmentů

Zjednodušený stavový diagram TCP



Další protokoly s IP

- ICMP
  - určený pro výměnu řídících informací infrastruktury – přesměrování, konektivita
  - nejpoužívanější příkazy PING a TRACEROUTE
- ARP a RARP
  - realizuje nalezení fyzické adresy zařízení (např. u Ethernetu je to MAC síťové karty) k IP adrese
  - RARP - umožňuje např. vzdálený boot počítače
- NTP – Network Time Protocol
  - Synchronizace hodin počítače

FTP protokol

- Vytváří se 2 spojení se serverem
  - Výměna příkazů - port 21
    - Na tomto portu server "naslouchá"
    - Textově pomocí Telnetu (jako modul nebo externě)
  - Přenos dat – inicializace na portu 20, využívá se TCP port klienta
- Složitá analýza odpovědí od různých FTP serverů
  - Pro zajímavost - „zdrojáky" FileZilla (navíc je i Server)
- Přenosové režimy
  - Proudový mód – celý soubor je sekvencí bajtů, na konci EOF
  - Blokový mód – v bloku hlavička s velikostí bloku, možnost přenosu od určitého místa
  - Komprimovaný mód – komprese obdobná RLE

HTTP protokol a přenos dat

- Identifikace cílového počítače pomocí URL (Uniform Resource Locator)
  - scheme://username:password@domain:port/path?query_string#fragment_id
- Využívá se standard MIME (Multipurpose Internet Mail Extensions) pro přenos ne-textových informací (RFC822)

- Příkaz GET
  - Předá serveru URL s příp. parametry v textové podobě
  - Max. délka není v RFC2616 specifikována
    - IE6 – 2083 bajtů
    - FF – omezen na 64k při zobrazení, teoreticky i více
    - Opera – 190k testováno
    - Apache – starší verze cca 4k, oficiálně 8k
    - IIS – defaultně 16k
    - server by měl příp. vrátit chybu 414 „Request-URI Too Long“
  - Data oddělena ‘?’ od adresy, položky navzájem ‘&’, typicky jsou použity páry „klíč=hodnota“
- Příkaz POST
  - Slouží k zasílání dat na server (i v binární podobě díky MIME)
  - Velikost dat často omezena serverem (např. PHP defaultně 2MB)
  - Data zakódována podle uvedeného ENCTYPE v hlavičce

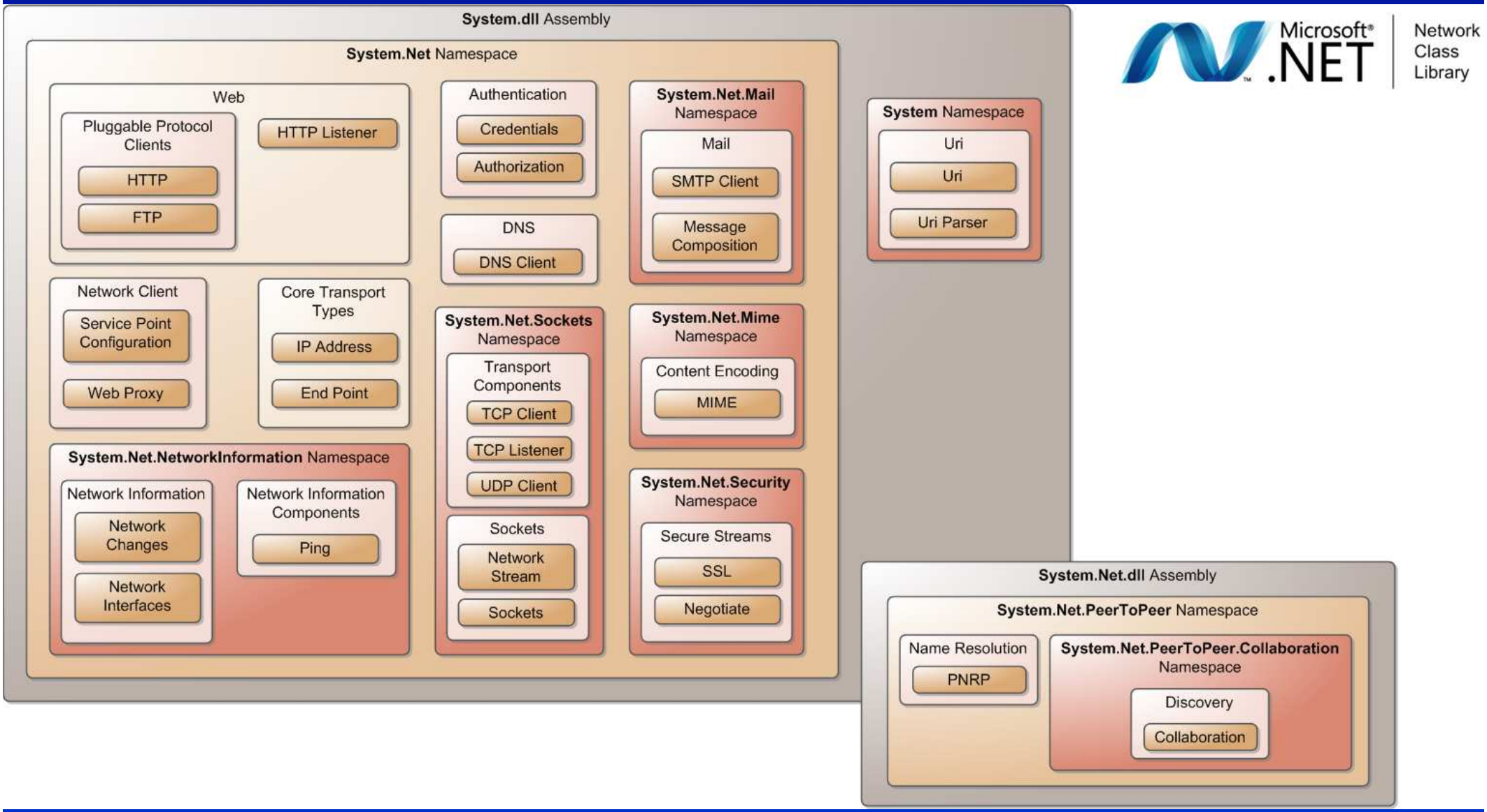
Využitelné třídy v .NET2+

- **Socket**
  - základní komunikační funkcionalita
  - umožňuje synchronní i asynchronní přenosy
- **UdpClient**
  - umožňuje odesílat i přijímat UDP pakety
- **TcpListener** + **TcpClient**
  - Listener slouží pro příjem požadavků od „Clientů“
- Součástí jmenného prostoru **System.Net.Sockets** je také např. IrDAClient a IrDAListener

Využitelné třídy v .NET2 - II

- Součástí jmenného prostoru **System.Net** jsou i „vysokoúrovňové" třídy
  - **HttpListener** – základní funkcionalita web serveru
  - **WebClient** – realizuje požadavek na www server (odesílání i příjem)
  - **FtpWebRequest** – požadavek na FTP server
- **System.Net** obsahuje i pestrou paletu „servisních" tříd
  - **IPAddress** a **IPEndPoint** (IP adresa + port)
  - **Dns** – informace z DNS serverů, typicky **Dns.GetHostByName**, **Dns.GetHostName** a **Dns.Resolve** – pozor, některé jsou „obsolete" a v .NET2+ nahrazeny „lepšími"
  - **Cookie** a různé ...Credential a ...Permission

Network Class Library



Kostra UDP aplikace

- Vysílání = nastavit spojení a odesílat
- Příjem = spustit příjem (ideálně asynchronně)
  - Při příjmu dat zpracovat a opět čekat

```
...
using System.Net;
using System.Net.Sockets;

static void Main(string[] args)
{
  UdpClient u = new UdpClient();
  u.Connect(
    new IPAddress(new byte[] { 192, 168, 5, 208 }),
    1234);

  while (true)
  {
    ...

    String s = Console.ReadLine();

    byte[] data = Encoding.ASCII.GetBytes(s);
    u.Send(data, data.Length);
  }

  u.Close();
}
```

```
static void Main(string[] args)
{
  UdpClient u = new UdpClient(1234);
  u.BeginReceive(new AsyncCallback(recvData), u);

  while (true)
    ...
}

private static void recvData(IAsyncResult ar)
{
  UdpClient uu = ar.AsyncState as UdpClient;
  if (uu == null)  return;

  IPEndPoint ipe = new IPEndPoint(IPAddress.Any, 0);
  byte[] data = uu.EndReceive(ar, ref ipe);

  ...
  uu.BeginReceive(new AsyncCallback(recvData), uu);
}
```

Kostra TCP aplikace - klient

- Navázat spojení se serverem
- Ve vlákně čekat na data
  - Žádná data = spojení ukončeno serverem
- Odesílání synchronní

```
TcpClient klient = null;

// event: close
if (klient.Connected)
  klient.Close();

// event: connect
try
{
  klient = new TcpClient();

  IPEndPoint ipe = new IPEndPoint(address, port);
  klient.Connect(ipe);

  Thread clientThread = new Thread(
    new ThreadStart(HandleClientComm));
  clientThread.Start();
}
catch (Exception ex) ...

// event: send
NetworkStream str = klient.GetStream();
str.Write(data, 0, data.Length);
str.Flush();
```

```
// thread
private void HandleClientComm()
{
  NetworkStream clientStream = klient.GetStream();

  byte[] message = new byte[4096];
  int bytesRead;

  while (true)
  {
    bytesRead = 0;

    try
    {
      bytesRead = clientStream.Read(message, 0, 4096);
        // blocks until a client sends a message
    }
    catch (Exception ex)
    {
      Log(ex.Message);
      break;
    }

    if (bytesRead == 0)
    {
      // klient.Client.RemoteEndPoint was disconnected
      break;
    }

    string msg =
      Encoding.ASCII.GetString(message, 0, bytesRead);
    ...
  }
}
```

Kostra TCP aplikace - server

```
TcpListener tcpServer = null;
Thread thrListen = null;
List<TcpClient> lClients = new List<TcpClient>();

// event: close app
tcpServer.Stop();
thrListen.Abort();

foreach (var c in lClients)
  c.Close();

// event: start listening on port
tcpServer = TcpListener.Create(port);

thrListen = new Thread(
  new ThreadStart(ListenForClients));
thrListen.Start();

// thread - accept connections
private void ListenForClients()
{
  tcpServer.Start();

  while (true)
  {
    TcpClient clnt;

    try
    {
      clnt = tcpServer.AcceptTcpClient();
        // blocking - wait for connection
      lClients.Add(clnt);
    }
    catch (Exception ex)
    {
      break;
    }

    Thread clientThread = new Thread(new
      ParameterizedThreadStart(HandleClient));
    clientThread.Start(clnt);
  }
}
```

```
private void HandleClient(object client)  // thread for client
{
  TcpClient tcpClient = (TcpClient)client;
  NetworkStream clientStream =
    tcpClient.GetStream();

// connected tcpClient.Client.RemoteEndPoint;

  byte[] message = new byte[4096];
  int bytesRead;

  while (true)
  {
    bytesRead = 0;

    try
    {
      bytesRead = clientStream.Read(message, 0, 4096);
    }
    catch (Exception ex)
    {
      break;
    }

    if (bytesRead == 0)  // disconnected ?
    {
      lClients.Remove(tcpClient);
      break;
    }

    String msg = Encoding.ASCII.GetString(
      message, 0, bytesRead);
    Echo(msg, clientStream);  // stream write&flush
  }

  tcpClient.Close();
}
```

Odkazy na zdroje

- http://cs.wikipedia.org/wiki/Referen%C4%8Dn%C3%AD_model_ISO/OSI
- http://cs.wikipedia.org/wiki/TCP/IP
- http://en.wikipedia.org/wiki/Internet_Protocol
- http://cs.wikipedia.org/wiki/IPv4
- http://cs.wikipedia.org/wiki/IPv6
- http://stackoverflow.com/questions/417142/what-is-the-maximum-length-of-an-url
- http://stackoverflow.com/questions/266322/http-uri-get-limit
- http://en.wikipedia.org/wiki/Uniform_Resource_Locator
- http://en.wikipedia.org/wiki/Uniform_Resource_Identifier
- http://msdn.microsoft.com/en-us/library/system.net.dns.aspx
- http://msdn.microsoft.com/en-us/library/system.net.aspx

- https://code.msdn.microsoft.com/windowsapps/TCP-IP-Server-Client-0964d476

Přednáška 4 - **Tenký klient**

- HTTP protokol – základ internetu
- HTML a XHTML jazyk – popis www stránky
- WWW server
- Aktivní stránky – inteligence na pozadí
- PHP – nejjednodušší řešení
- ASP.NET – princip
- Programovací model prvků – Controls

HTTP protokol

- Textový protokol původně určený na výměnu hypertextových dokumentů ve formátu HTML
- Typicky port 80 protokolu TCP (443 pro HTTPS)
- Verze 1.1 v RFC 2616
- Uživatel (resp. klient SW) odešle serveru dotaz (typicky pomocí prohlížeče), odpovědí je textová hlavička popisující výsledek dotazu a jeho samotná data = požadovaný dokument
- Jedná se o bezstavový protokol – mezi dotazy neexistuje na serveru žádná souvislost/vazba

HTTP protokol - 2

- Metody pro získání dokumentu:
  - GET – nejčastější, případná data/parametry předána serveru jako parametry dotazu
  - HEAD – stejný jako GET, server nevrací tělo dokumentu, pouze hlavičku
  - POST – umožňuje poslat data z formuláře na stránce včetně binárních
- Příklad:

```
GET /wiki/Wikipedie HTTP/1.1
Host: cs.wikipedia.org
User-Agent: Mozilla/5.0 Gecko/20040803 Firefox/0.9.3
Accept-Charset: UTF-8,*
```

```
HTTP/1.0 200 OK
Date: Fri, 15 Oct 2004 08:20:25 GMT
Server: Apache/1.3.29 (Unix) PHP/4.3.8
X-Powered-By: PHP/4.3.8
Vary: Accept-Encoding,Cookie
Cache-Control: private, s-maxage=0, max-age=0, must-revalidate
Content-Language: cs
Content-Type: text/html; charset=utf-8
```

(X)HTML

- Značkovací jazyk pro popis WWW stránek ve formě hypertextových dokumentů
- Vychází z SGML (ISO 8879)
- Pro popis dokumentu užívá množinu značek a jejich atributů
- Značky uzavřeny v "lomených" závorkách <>
- Část dokumentu mezi značkami se nazývá prvek dokumentu (= element)
- Prvky se mohou vnořovat
- Značky jsou často párové, uzavírací značka je uvozena „lomítkem“

(X)HMTL – 2

- Poslední klasická verze HTML je 4.01 (12/1999, resp. ISO z 2000)
- Vývoj pozastaven ve prospěch XHTML (nyní v1.1 z 2001)
- XHTML 2 v rámci W3C pozastaven
- Vývoj HTML5 (stav 2014 – specifikace ve schvalování, stabilní množina společných vlastností, stále vývoj rozšíření)
- Povinná deklarace typu DTD (Document Type Description) určuje, jaké elementy mohou být využity (pro HTML 4.01 a XHTML)
- Struktura dokumentu je dána – DTD, kořenový element <html>, hlavička <head> a tělo dokumentu <body>

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN“ "http://www.w3.org/TR/html4/strict.dtd">
<html>
  <head>
    <title>Titulek stránky</title>
  </head>
  <body><!-- komentar -->
    <h1>Nadpis stránky</h1>
    <p>Toto je tělo dokumentu</p>
  </body>
</html>
```

(X)HTML - 3

- Druhy značek:
  - strukturální – rozvrhují strukturu dokumentu – odstavce <p>, nadpisy <h1>, bloky , ...
  - popisné (sémantické) – popisují obsah elementu, usnadňují strojové zpracování – příklad <title>, <address>, v HTML5 přibývá řada dalších
  - stylistické – určují vzhled elementu (např. <b> pro tučný text) – doporučuje se nepoužívat
- Režimy práce moderních prohlížečů
  - standardní – dodržují standard dle DTD, netolerují syntaktické prohřešky
  - "quirk" – prohlížeč si domýšlí při chybě apod.

(X)HTML – 4

- XHML 1.1 – stále aktuální standard
  - dokument respektuje požadavky XML standardu při zachování max. kompatibility s HTML
  - všechny prvky párové, uzavřené, nesmí se „křížit“
  - atributy v uvozovkách

- CSS – kaskádové styly
  - oddělení vzhledu (CSS) od obsahu (XHTML)
  - typicky v odděleném souboru, společný více stránkám
  - popisují vzhled prvků dokumentu
  - možno i přímo v dokumentu
  - hierarchicky členěné, inspirace dědičností

(X)HTML – 5

```
p, body, td, th, ul, li
{
  font-family: Arial,Helvetica,Geneva,Swiss,SunSans-Regular,sans-serif,ms-serif;
  font-size: small;
}

body
{
  padding: 0px;
  margin: 0px;
  background-image: url('/images/design/bg_svetlemodra.gif');
}
```

```
.blok-top-best
{
  text-indent: -10px;
  margin-left: 10px;
}

.blok-top-best a
{
  color: #000000;
}
```

```
<!DOCTYPE html .. DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>FEL Plzen</title>
  <meta http-equiv="Content-Type" content="text/html; charset=windows-1250" />
  <link rel="stylesheet" type="text/css" href="/main.css" />
  <link rel="stylesheet" type="text/css" href="/main_print.css" media="print" />
</head>
<body>
  <div align="center"><!-- tento DIV tu musi byt kvuli centrovani v Gecko !! -->
....

  <div class="blok-top-best"><a href="fakulta.aspx">O fakultě</a>
  </div>
</body>
</html>
```

(X)HTML – 6 – standard **HTML5**

- W3C uvedla finální verzi v říjnu 2014
- Vylepšení a změny
  - nové prvky formulářů
  - přímé přehrávání audio/video bez pluginů
    - náhrada Flash přehrávačů
  - nové tagy a lepší analýza sémantiky
  - nové styly CSS (CSS3+)
  - zjednodušené hlavičky a „meta" informace
  - nové nativní rozhraní pro lokální ukládání dat
    - IndexedDB, dříve WebSimpleDB
  - objekt Canvas
    - prostředí pro (animovanou) 2D grafiku
    - potenciální náhrada animací Flash nebo Silverlight
- V prohlížečích různá podpora dle vývojových verzí a „chuti" programátorů
- Zajímavý agregátor článků např. na http://www.html5.cz/

WWW server

- Vyřizuje požadavky předané HTTP protokolem = odesílá dokumenty
- 2 druhy přístupu k dokumentům
  - statický
    - dokumenty odpovídají souborům na disku
    - prakticky se řeší jen oprávnění přístupu k souboru
  - dynamický
    - podle předpisu generuje výsledný dokument
    - typicky je součástí předpisu část "statická" (= HTML kód) a výkonný algoritmus (zapsaný v programovacím jazyku)

WWW server - 2

- URL (Uniform Resource Locator) - jednoznačné určení pozice dokumentu na internetu
  - obsahuje protokol, doménové jméno, port, specifikaci souboru a parametry
  - některá pole jsou nepovinná, příp. implicitní
- Nejrozšířenější WWW servery
  - Apache - multiplatformní
  - IIS - Windows XP = IIS 5.1, Server 2003 = IIS 6, Server 2008 = IIS 7 (.5), Server 2012 = IIS 8
  - Sun Java System Web Server - častý pro hostování Java aplikací/řešení

Aktivní obsah

- CGI
  - libovolný spustitelný kód na dané platformě (EXE, různé skripty Shellu)
  - nejčastěji C/C++, Perl, Python, ...
- PHP
  - v amatérské oblasti asi nejrozšířenější, OpenSource
- ASP a ASP.NET
  - podpora Microsoftu a omezení prakticky jen na Windows (na Linuxu iniciativa projektu Mono – dříve Novell, pak Xamarin, MS)
  - klasické ASP podobné PHP – už se nepoužívá
  - ASP.NET řeší většinu problémů pomocí objektového přístupu
- Java
  - free nástroje, multiplatformní vývoj i běh
  - na objektovém základě různé principy – JSP, Servlety, ...

CGI aktivní stránky

- Windows
  - většinou DLL z C/C++
- Linux
  - každý spustitelný skript má dáno, jak jej lze spustit
  - stačí vhodně nastavit Apache
  - PERL - objekty/moduly na „cokoliv“
    - např. www.cpan.org - Comprehensive Perl Archive Network

```
#!/usr/bin/perl -w
#
use File::Find;
use File::Path;
use File::Basename;
use Cwd;
if (&Portability::MyOS eq 'Win32')
{
  eval 'use Win32::TieRegistry( Delimiter=>"/", ArrayValues=>0 );';
}

$SIG{__WARN__} = 'warn_log';
$SIG{__DIE__} = 'die_log';

$OUT_TEXT = 0;
$OUT_DOC = 1;
$OUT_HTML = 2;

$main::home_url = "http://sitescooper.tsx.org";
$main::debugdiffs = 0;  # set to 1 to break after diffing
```

PHP

- Dostupný i ve zdrojové formě – existuje prakticky pro každou platformu, možnost předložit podle požadavků
- Syntaxe podobná C/C++, proměnné začínají $, členské prvky objektů ->
- Stále rostoucí balík funkcí pro realizací různých činností (např. DB konektivita, ...)
- Nevyrovnaná kvalita modulů, různé předávání parametrů (podle zvyklostí autorů modulu)
- ! interpretovaný – tisíce řádků kódu v přilinkovaných knihovnách
- ! použití proměnné = deklarace proměnné !!

- Dynamický typ = datový typ proměnné se určí při přiřazení
- Pole heterogenní = mohou obsahovat "cokoliv", podobně i indexy
- Automatická konverze typů => prakticky žádná typová kontrola

- Funkčnost objektů výrazně posílena od verze 5.x.x - při přechodu z 4.x možno očekávat problémy s nekompatibilitou

PHP - 2

- Pro rozsáhlejší projekty se používají mechanismy pro oddělení vizuální podoby (HTML) a aktivního obsahu (PHP)
  - příklad Smarty – dříve podprojekt PHP, nyní www.smarty.net
  - HTML kód je v tzv. šabloně, která je pomocí objektů frameworku Smarty navázána na konkrétní PHP stránku
  - vhodně se dělí funkce webdesignéra a programátora
- Pro přístup k datům lze použít univerzální objektový přístup typu ADODB, který obsahuje univerzální rozhraní pro různé DB a je inspirován ADO.NET od Microsoftu

PHP - 3

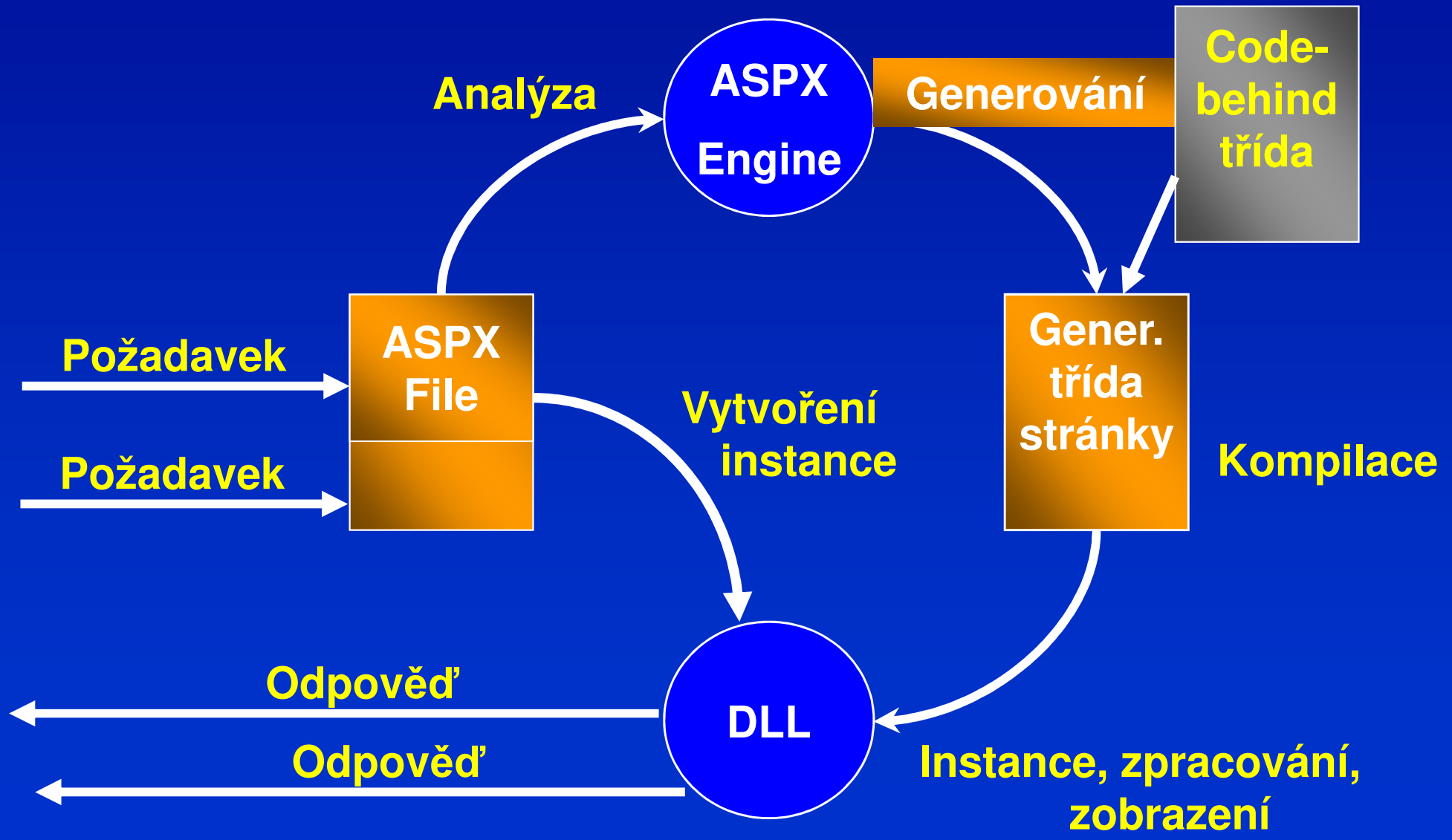
```php
<?php
$link = mysql_connect("localhost", "jmeno", “heslo") or die("Nelze se připojit");

$res = mysql_query("SELECT * FROM tbl WHERE id > 0") or die("Err: " . mysql_error());

while ($row = mysql_fetch_array($res, MYSQL_NUM))
{
    printf ("ID: %s  Jméno: %s", $row[0], $row[1]);
}

mysql_close($link);
?>
```

```php
<?php
include('adodb.inc.php'); # load code common to ADOdb

$conn = &ADONewConnection('access'); # create conn.
$conn->Connect('northwind'); # connect to MS-Access

$rs = $conn->Execute('SELECT * FROM products');
if (!$rs)
  print $conn->ErrorMsg();
else
  while (!$rs->EOF)
  {
    print $rs->fields[0].' '.$rs->fields[1].'<BR />';
    $rs->MoveNext();
  }

$rs->Close(); # optional
$conn->Close(); # optional
?>
```

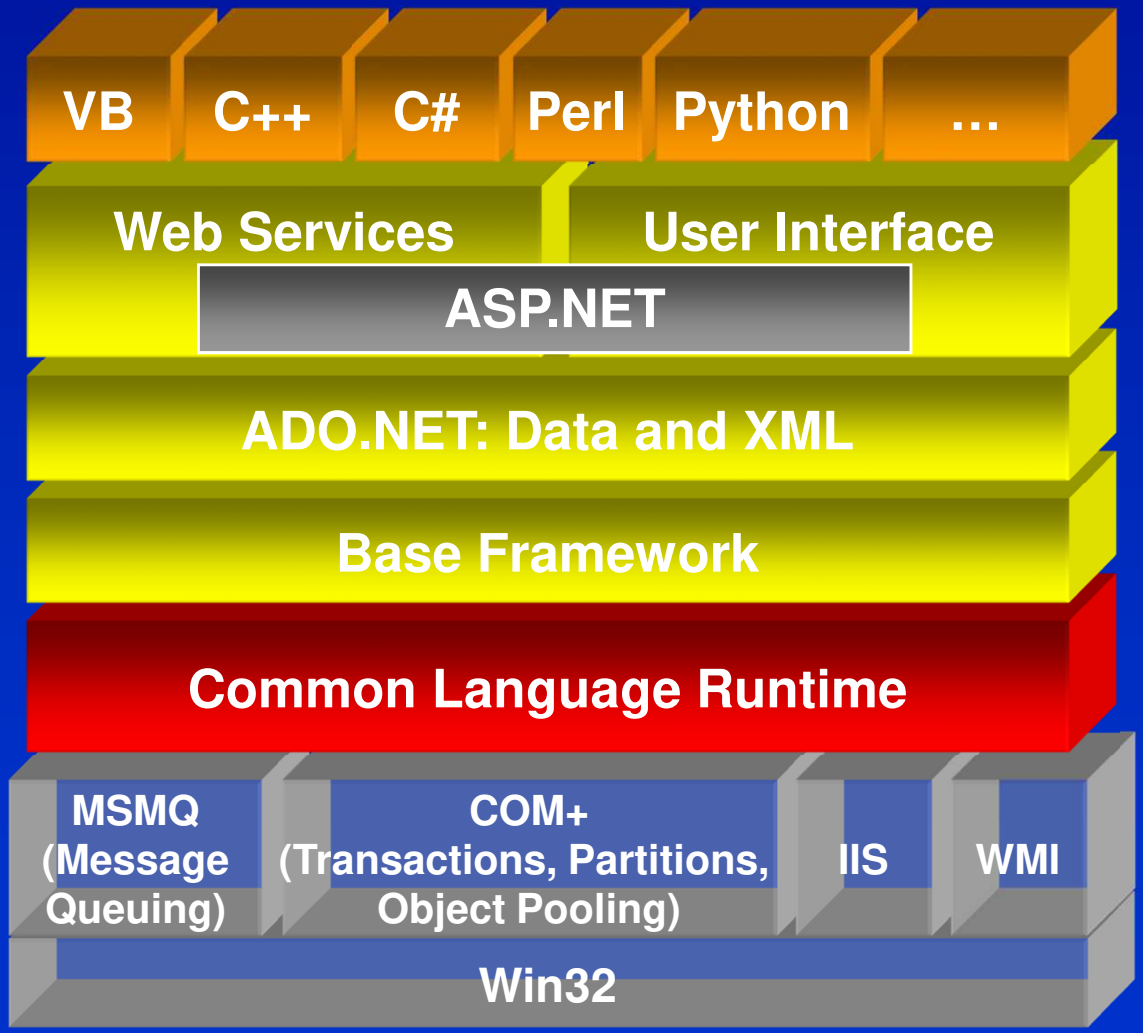
ASP.NET

- Vystavěno nad objektovým modelem .NET frameworku
  - pro přístup k datům a zdrojům počítače stejné třídy (objekty) jako běžné aplikace nad .NET
  - inteligenci (kód) lze psát ve všech jazycích .NETu (C#, VisualBasic, ...)
- Možno rozdělit na soubor vizuální části stránky + kód realizující třídy (CodeBehind) nebo vše v jednom souboru
- Typická přípona stránky je .aspx (resp. as?x)

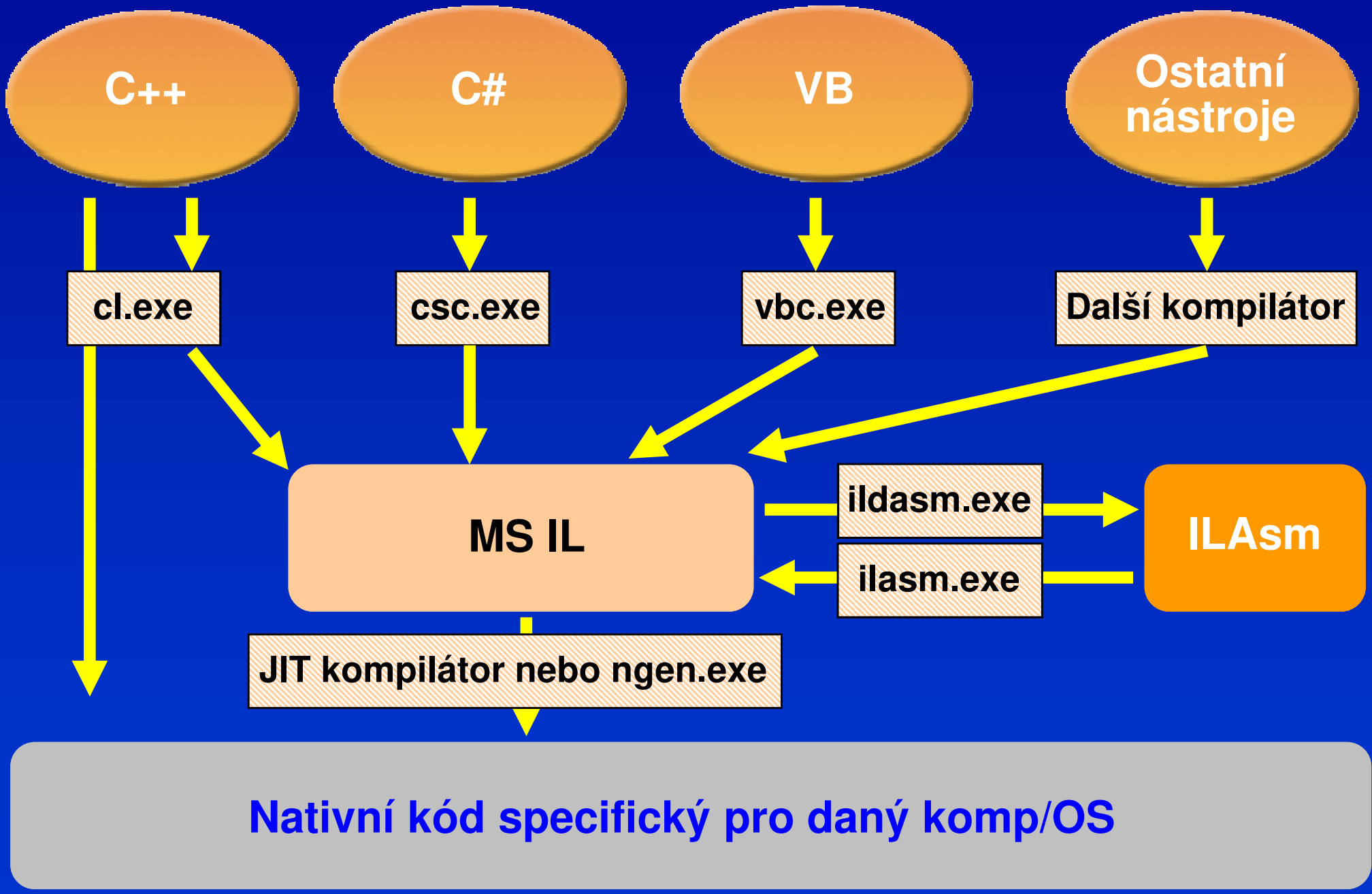
ASP.NET - 2


Analýza
ASPX Engine
Generování
Code-behind třída
Požadavek
Požadavek
ASPX File
Vytvoření instance
Gener. třída stránky
Kompilace
Odpověď
Odpověď
DLL
Instance, zpracování, zobrazení

ASP.NET - 3


VB
C++
C#
Perl
Python
...
Web Services
User Interface
ASP.NET
ADO.NET: Data and XML
Base Framework
Common Language Runtime
MSMQ (Message Queuing)
COM+ (Transactions, Partitions, Object Pooling)
IIS
WMI
Win32

ASP.NET - 4


C++
C#
VB
Ostatní nástroje
cl.exe
csc.exe
vbc.exe
Další kompilátor
MS IL
ildasm.exe
ilasm.exe
ILAsm
JIT kompilátor nebo ngen.exe
Nativní kód specifický pro daný komp/OS

ASP.NET - 5

- WebForms – obdoba WinForms (tlustého klienta), tedy:
  - Jednotlivé prvky (web)UI jsou reprezentovány objekty (Button, CheckBox, ...)
  - Do www jsou implementovány mechanismy událostního programování známé z běžných aplikací
  - Události jsou
    - generovány klientem
    - zpracovány na serveru
    - zpracování události znamená komunikaci po síti
- Automatické posílání události pro prvky se zapíná či vypíná vlastností AutoPostBack pro stránku nebo prvek
  - související vlastnost **Page.IsPostBack** stránky
    - false - první zobrazení stránky
    - true – zpracování události
- Pokud není aplikace určena pro intranet, je vhodné s událostmi šetřit !!

- Http je bezestavové
  - stav prvků se neukládá na serveru
  - položka "ViewState" cestuje na klienta a zpět jako skryté pole formuláře
    - možno ukládat na serveru a přenášet jen "ID" (např. cookie)

ASP.NET - 6

- Podporováno “XCOPY” nasazení
  - odpadá nutnost registrace komponent a nutnost restartu služeb/serveru
- Podporováno pro všechny součásti
  - web stránky, web služby
  - zkompilované komponenty (DLL)
  - konfigurační soubory
- Aktualizace aplikací za provozu
  - nakopírovat nové DLL na místo původního
  - aplikace použijí nové DLL počínaje příštím klientským požadavkem

- Konfigurace uložena jako XML soubor spolu se stránkami
  - název souboru **web.config**
- Obsahuje všechna nastavení ASP.NET
  - autentizace, kompilace
  - trasování, ladění, chybové stránky
  - moduly, handlery, …
- Jakékoliv nastavení doporučeno v sekci <appSettings>
  - spojení do databází, cesty k datovým adresářům, ...
- Rozšiřitelná konfigurace v rámci XML pravidel
  - možnost přidávat vlastních sekce dat

ASP.NET - 7

- Vlastní formulářová autentizace
  - jednoduché použití, přihlašovací tiket ve formě šifrovaného cookie
  - vlastní přihlašovací stránka (žádný pop-up)
  - ověření proti databázi, proti konfiguračnímu souboru web.config, …
- Autentizace Basic, Digest, NTLM
  - využívá IIS
  - jméno/heslo ověřeno proti AD/SAM
- Modul pro passport autentizaci
  - přístupný Microsoft-Passport profil

- Často řešený problém
  - aplikace si musí pamatovat obsah "nákupního košíku" uživatele
  - http je bezestavový – musíme použít cookies a evidovat relace (sessions)
- ASP.NET umí stav Session uložit
  - na webovém serveru (RAM, stejný proces)
  - na ASP.NET State Server (RAM, jiný proces)
  - na SQL Serveru

ASP.NET - WebControls

- Třída Page
  - reprezentuje celou webovou stránku
  - kontejner pro ostatní objekty
- Třída Button, CheckBox
- Třída Label
- Třída ListBox
- Třída Calendar
- Třída Repeater – pro opakovaná data, typicky tabulky
- Třída DataGrid
  - zobrazuje obsah databázových tabulek
  - umožňuje editaci dat a odeslání změn do databáze

- Definovat vlastní třídu pro web control
  - potomek System.Web.UI.WebControls.WebControl
- Definovat vlastnosti třídy
  - uživatelé mohou vlastnosti nastavovat přímo ve Visual Studiu .NET
  - a potom...
    - složit objekt z jednodušších objektů (composite web controls)
    - ... nebo implementovat kompletní (X)HTML výstup
      - přepsání metody Render

ASP.NET - příklad 1

```
<%@ Page Language="C#" MasterPageFile="~/intra.master" CodeFile="data.aspx.cs"
  Inherits="data_aspx" Title="Data" Trace="true" %>
<asp:Content runat="server" ContentPlaceHolderID="bodyPlaceHolder">
<h1>Datova administrace</h1>
<table border="0" cellpadding="2" cellspacing="0" class="tbl">
  <tr>
    <td>
      <asp:ListBox runat="server" Rows="1" ID="lbOperace">
        <asp:ListItem Text="Import dat komplet" Value="imp_komplet" />
        <asp:ListItem Text="Import posledni mesic" Value="imp_last" />
      </asp:ListBox>
    </td>
    <td align="center"><asp:Button runat="server" OnClick="btnDoIt_Click" Text="Do" /></td>
  </tr>
  <tr>
    <td colspan="2"><asp:FileUpload ID="fuSoubor" runat="server" /></td>
  </tr>
</table>
<pre>
<asp:Literal ID="ltrOutput" runat="server"></asp:Literal>
</pre>
</asp:Content>
```

ASP.NET příklad 2

```
using System;
using System.Data;
...
public partial class data_aspx : fel.WebControls.BasicPage
{
  void Page_Load(object sender, EventArgs e)
  {
    Page.Trace.Write("Page_Load", Page.Request.Path);
    if (!IsPostBack)
    {
//      System.Diagnostics.Debug.WriteLine("Toto je post-back");
    }
  }
  public void btnDoIt_Click(Object sender, EventArgs e)
  {
    switch (lbOperace.SelectedValue)
    {
      case  "imp_komplet": ImportAll(fuSoubor); break;
...
    }
  }
  private void ImportAll(FileUpload fu)
  {
    if (!fu.HasFile)
    {
      ltrOutput.Text = "Neni predan soubor, konec ..."; return;
    }

    using(StreamReader rd = new StreamReader(fu.FileContent))
    {
      if (rd.BaseStream.CanRead)
      {
        using(MySqlConnection conn = _fel.GetConnection())
        {
          conn.Open();
          using(MySqlCommand cmd = conn.CreateCommand())
          {
            cmd1.CommandText = @"SELECT id ...
            object o = cmd1.ExecuteScalar();
          }
          conn.Close();
        }
      }
    }
    ltrOutput.Text = sb.ToString();
    return;
  }
}
```

ASP.NET - životní cyklus stránky

- Zpracování ASP.NET stránky na serveru je řízeno událostmi a reagujícími metodami od požadavku po výsledně generovaný (X)HTML kód
  - OnInit (událost Init) - inicializace objektů uvnitř stránky
  - LoadViewState - načtení stavových informací prvků
  - LoadPostData - zpracování dat formulářů, aktualizace vlastností
  - **OnLoad** (Load) - hlavní výkonná část, data aktuální, automaticky přihlášen
  - PostData Changed - oznámení o změně dat proti původním
  - PostBack Event - obsluha **událostí prvků** (OnClick, ...)
  - **LoadComplete** – post-back a view-state data zpracována, vhodné pro vykreslení podle stavu, nutno přihlásit Event v OnInit
  - OnPreRender (PreRender) - poslední možnost změny vlastností
  - SaveViewState - uložení stavu prvků (__VIEWSTATE - Base64)
  - Render - do výstupního proudu zapsána (X)HTML podoba stránky
  - Dispose - úklid objektů
  - OnUnload (UnLoad)

ASP.NET - MVC

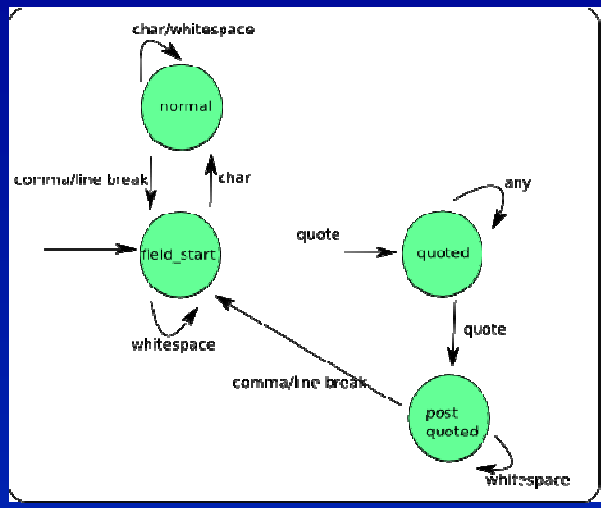
- Návrhový vzor "Model-View-Controller"
  - Striktní oddělení logiky dat od výstupní reprezentace
  - Model = výpočetní logika (včetně DB operací)
    - Vstupem jsou parametry funkcí objektu
  - View = popis, jak jsou data zobrazena
    - HTML šablona a speciální tagy (proměnné, cykly, ...)
    - Šablony možno vnořovat (obdoba Include)
    - Může obsahovat i trochu výpočetní logiky (kontrola zadaných vstupů apod.)
  - Controller = mapuje cestu dat mezi Model a View a zpracovává vstupy od uživatele
  - (Router) = podle URL požadavku vybere příslušný Controller
- VS strukturovaně podporuje jednotlivé komponenty
    - Wizards, Templates, ...

Přednáška 5 - prezentace dat

- Standardní formáty dokumentů
- Generování dokumentů
  - tlustý klient
  - tenký klient - webové aplikace
- Generování obrázků

- .Net pro mikrokontroléry – rychlý přehled
  - Micro.Net Framework
  - následovník – TinyCLR OS
  - alternativa - NanoFramework

Formát CSV



- Nejjednodušší formát pro tabulková data
- "Comma Separated Values" = hodnoty oddělené čárkou
  - prakticky se používá středník
- Řádky dat odpovídají řádkům souboru
- Textové položky bývají uzavřeny do uvozovek
- Problémy s načítáním
  - v případě obecných textových dat nelze rozdělit podle oddělovače (může být součástí textu)
  - nutný stavový automat, který respektuje text v uvozovkách a oddělovač
- V .NET lze využít přímo MS JetEngine, který umí zpracovat CSV jako DataTable apod.

```
using (OleDbConnection dbConn = new OleDbConnection(
  "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=ADRESAR;Extended Properties='text;HDR=Yes;FMT=Delimited'"))
{
  dbConn.Open();
  using (OleDbCommand cmd = dbConn.CreateCommand())
  {
    cmd.CommandText = "SELECT teplota FROM teploty.csv WHERE stanice = ?";  // parametry urcuje poradi
    cmd.Parameters.AddWithValue("stanice", 1119);

    using (OleDbDataReader rdr = cmd.ExecuteReader())
    {
      while (rdr.Read())
      {
         float f = rdr.GetFloat(0);
         ...
```

Formát XML

- Možno formálně validovat obsah
- Možno vytvořit vlastní strukturu dokumentu a podle DTD (Document Type Definition) validovat
- Možno se na data dotazovat pomocí XPath = dotazovací jazyk
  - definuje se výběrová množina uzlů dokumentu splňující požadovanou podmínku
  - typicky se jednotlivé uzly iteračně procházejí
  - možno se odkazovat i relativně v XML dokumentu
- V rámci vyšších jazyků různě rozsáhlá podpora
  - PHP - např. objektová knihovna PHPXPATH
  - .NET má funkce pro efektivní vytváření a čtení XML souborů včetně XPath výrazů
  - .NET má přímo možnost načíst DataSet z XML

Načítaní pomocí XPath - příklad

```
XPathDocument xTreeDoc = new XPathDocument(..soubor..);
XPathNavigator navig = xTreeDoc.CreateNavigator();
if (navig != null)
{
  XPathNodeIterator iter = navig.Select("//root/item");  // XPath vyraz
  if ((iter != null) && (iter.Count != 0))               // něco "se naslo"
  {
    while (iter.MoveNext())                 // dopredne prochazeni "kolekce vysledku"
    {
      int iVal = iter.Current.ValueAsInt;
      String sAttribDate = iter.Current.GetAttribute("date", "");
      ...
    }
  }
}
```

- Alternativně lze vyjít z XmlDocument:

```
XmlDocument doc = new XmlDocument();
try { doc.Load(filename); }
catch (FileNotFoundException ex) { ... }

XPathNavigator navig = doc.CreateNavigator();
XPathNodeIterator iter = navig.Select("//parametry");
if (iter.MoveNext())
{
   XPathNodeIterator subiter = iter.Current.SelectChildren(XPathNodeType.Element);
   while (subiter.MoveNext())     // iteracni prochazeni polozek vysledku
```

Formát PDF (Adobe Acrobat)

- Nejrozšířenější formát pro dokumenty
- Vychází z PostScriptu + vnitřní ZIP komprese
- Existuje řada knihoven pro vytváření
  - např. hledat "PDF" na SourceForge pro nekomerční projekty (např. iTextSharp – klon Javového iTextu)
  - velká škála komerčních komponent
- Princip tvorby společný
  - vytvoří se dokument PDF
  - pomocí "primitiv" se vytváří obsah (text, line, font, ...)
  - výsledek se vyrenderuje do PDF formátu
- Dostupné knihovny i pro načtení a analýzu
- Problém s vykreslováním
  - není free modul pro Windows (aktuálně Sumatra PDF ?)
  - možnosti - Acrobat SDK (neaktuální), GhostScript engine

Grafické formáty

- Rastrové
  - různé komprese - žádná (BMP), bezztrátová, ztrátová
  - různé barevné hloubky - 256 (GIF), 16-, 24- nebo 32-bitové barvy
  - možnost průhlednosti (musí podporovat vykreslující aplikace)
  - namátkou - GIF, JPG, PNG, BMP, TIFF, TGA, ...
- Vektorové
  - pro některé typy obrázků úspornější
  - nevznikají artefakty při vykreslování
  - rozsáhlé obrázky náročnější na HW výkon
  - typicky - WMF, (E)PS, 3DS, VRML, ...

Ostatní typy souborů

- Excel
  - umí zpracovat XML i CSV
  - nativní formát XLS
    - práce s tabulkami, zápis do souřadnic
    - v PHP několik knihoven různé kvality
    - v .NET existují free i komerční moduly pro export/import
    - v .NET je možno importovat COM objekt Excel
      - pozor na stejnou verzi na cílovém počítači
      - pro ASP.NET musí být na serveru
      - Office objekty nejsou typicky určeny pro multi-task/thread prostředí (tedy pozor na ASP.NET)
- Word
  - nejvhodnější využít RTF formát = čistý text s formátovacími povely

Generování souborů

- Textové soubory možno pomocí fprintf
- Binární formáty pomocí write (do souboru)
- V C++ a C# možno použít objekty Stream
  - efektivnější využití
  - řada vyšších funkcí či objektů vyžaduje Stream
  - např. MemoryStream je v podstatě abstraktní vrstvou nad polem bytů v paměti

Generování souborů na WWW

- Vytvoření souboru na disku a poskytnutí odkazu na něj
  - musí existovat oblast (adresář), kam má právo zápisu služba/démon realizující WWW server
  - \+ výhodné z hlediska výkonu - data jsou vygenerována jen jednou
  - – nutno ošetřit případ, kdy existuje link a není soubor
  - – nutno "uklidit" nepotřebné soubory
- Generování on-the-fly
  - pro každý požadavek se vygeneruje "nový" obsah souboru a odešle v nativním formátu klientovi (prohlížeči)
  - \+ vždy k dispozici aktuální obsah souboru
  - – nutno vyřešit cachování, jinak může snadno dojít k zahlcení serveru
- Optimální kombinovat
  - "trvalky" jako soubor
  - aktuální data generovat přes vhodně "hlubokou" cache

Hlavičky a jejich data

- Formálně popsány v RFC2616 (HTTP protokol)
- Možnosti Content-Type:
  - text/html; charset=windows-1250
  - application/xhtml+xml; charset=utf-8
  - image/jpeg
  - application/pdf
  - application/octet-stream
- Ovládání Cache
  - Cache-Control: private
- Datum, server, cookie ...
- Content-Length, Referer

Realizace v PHP

- Nastavení ContentType se provádí funkcí Header, která obecně umí vkládat informace do hlavičky
- Vytvoření grafické plochy prostředky knihovny GD
  - ze souboru - ImageCreateFromJpeg
  - prázdný - ImageCreate nebo ImageCreateTrueColor
- Kreslení
  - ImageColorAllocate - tvorba "barvy"
  - ImageFilledRectangle - obdélník
  - ImageString
  - ImageLine
- Výstup do prohlížeče funkcí ImageJPEG
- Důležité
  - nezapomenout uvolnit paměť pomocí ImageDestroy
  - pozor na chybová hlášení, nutno je odchytit, v generovaném souboru nemají co dělat
  - výstup do GIF obecně nebyl podporován kvůli patentovým sporům
  - při použití funkcí z GD2 nutno předem otestovat existenci

Realizace v PHP - příklad

```
<?
Header("Content-type: image/jpeg");

$obrazek = false; $str = "OK";

if (IsSet($_GET["pict"]))                          // predany parametr
{
  $strFile = "./files/" . $_GET["pict"];           // fyzicke umisteni souboru
  if (NULL == ($aFile = @GetImageSize($strFile))) // vlastnosti obrazku v souboru
    $str = $_GET["pict"] . " not found";
  else
    switch($aFile[2])                              // nastaven podle typu obrazku
    {
      case    2:  $obrazek = @ImageCreateFromJPEG($strFile); break;
      case    3:  $obrazek = @ImageCreateFromPNG($strFile); break;
      default:    $str = "Unknown Image format";
    }
}
else
  $str = "Parameter error !!";

if (function_exists(imagecreatetruecolor))
    $main = imagecreatetruecolor(128, 192);
else
    $main = imagecreate(128, 192);

imagefilledrectangle($main, 0,0,128,192, imagecolorallocate ($main, 0x92, 0xa4, 0xca));

... %
```

Realizace v PHP - příklad II

```
...
if ($obrazek)
{
  if (($aFile[0] == 128) && ($aFile[1] == 192))
    imagecopy($main, $obrazek, 0, 0, 0, 0, 128, 192);
  else
  {
    ... // prepocet rozmeru

    if (function_exists(imagecopyresampled))
      imagecopyresampled($main, $obrazek, $x, $y, 0, 0, $w, $h, $aFile[0], $aFile[1]);
    else
      imagecopyresized($main, $obrazek, $x, $y, 0, 0, $w, $h, $aFile[0], $aFile[1]);
  }
  ImageDestroy($obrazek);
}
else
{
  $tc  = imagecolorallocate ($main, 0, 0, 0);
  imagestring ($main, 2, 5, 5, $str, $tc);
  imageline($main, 0, 20, 127, 191, $tc);
  imageline($main, 127, 20, 0, 191, $tc);
}

ImageJPEG($main);
ImageDestroy($main);
?>
```

Vykreslení grafiky v .NET Forms

- Obsluha zprávy OnPaint
  - jako parametr dostává PaintEventArgs
  - jeho členem je objekt Graphics, do kterého se již normálně kreslí pomocí funkcí a objektů GDI+
- Zpráva se vyvolá automaticky při odkrytí okna nebo "ručně" pomocí Invalidate
- Nový vlastní Control
  - možno zdědit z nějakého standardního
  - zdědit pouze základní třídu Control a ostatní doplnit ručně včetně všech specifik chování

Realizace v ASP.NET

- Lze použít "běžnou" aspx stránku
- Nesmí se vygenerovat jiný obsah než ten, který má obsahovat výsledný dokument
- Před renderováním je třeba nastavit potřebný ContentType
  - defaultně je text/html nebo podobný odpovídající www stránkám

Handler v ASP.NET

- Pouze jeden soubor - **neco.ashx**
- Zjednodušená režie generování stránky
- Objekt stránky dědí rozhraní IHttpHandler
- Využitelná prakticky jen událost ProcessHandler = obdoba OnLoad
- Pracuje se přímo s objektem Response, resp. s předaným context (kde je také Request pro zjištění parametrů)

- Typicky nutno:
  - nastavit Response.ContentType (textový řetězec) – def. „text/txt"
  - využít výstup do context.Response.OutputStream
  - alternativně nastavit práci s Cache

```
context.Response.Cache.SetCacheability(HttpCacheability.NoCache);
```

```
<%@ WebHandler Language="C#" Class="Handler" %>
```

Příklad v ASP.NET

```
...
public void ProcessRequest(HttpContext context)              // pozor, nejsou pouzity using a vsude Dispose !!
{
  NameValueCollection cGet = context.Request.QueryString;    // parametry z URL
  ...
  Bitmap bmp = new Bitmap(width, height, System.Drawing.Imaging.PixelFormat.Format24bppRgb);
  Graphics gr = Graphics.FromImage(bmp);                     // kreslici plocha napojena na bitmapu

  Font fnt = new Font("Arial", 7);                           // font pro psani do grafiky

  gr.FillRectangle(Brushes.White, 0, 0, bmp.PhysicalDimension.Width - 1, bmp.PhysicalDimension.Height - 1);
  gr.DrawRectangle(Pens.Black, 0, 0, bmp.PhysicalDimension.Width - 1, bmp.PhysicalDimension.Height - 1);
  gr.DrawLine(Pens.Black, 0, max_temp, width - 1, max_temp);

  for (int i = 0; i < width; i += 60)                        // vykresleni mrizky i s popisem
  {
    gr.DrawLine(Pens.Black, i, 0, i, sizes.height - 1);
    if (i == 0)                                              // pro prvni se popis negeneruje
      continue;
    gr.DrawString(String.Format("{0:D02}:{1:D02}", i / 60, i % 60), fnt, Brushes.Black, i + 1, 0);
  }
  ...
  SqlDataReader rdr = cmd.ExecuteReader();                   // pristup k datum, nekde vytvoren SQL CMD
  while (rdr.Read())
  {
    ...
    gr.FillRectangle(Brushes.Black, iPrevTime, sizes.max_temp - 1, iTime - iPrevTime, 3);
  }

  if (fnt != null) fnt.Dispose();                            // uvolneni fontu

  context.Response.ContentType = "image/gif";                // typ souboru
  context.Response.Cache.SetCacheability(HttpCacheability.NoCache);     // rizeni cache
  bmp.Save(context.Response.OutputStream, System.Drawing.Imaging.ImageFormat.Gif);  // "vysypat" do prohlizece
}
```

Zajímavé odkazy

- http://sourceforge.net/projects/phpxpath/
- http://www.phpclasses.org/browse/package/1919.html
- http://www.ietf.org/rfc/rfc2616.txt
- http://cz.php.net/header
- http://pgl.yoyo.org/http/browser-headers.php
- http://interval.cz/clanky/hlavicky-headers-v-php/
- http://www.fi.muni.cz/usr/brandejs/P005/uvodhttp.html
- http://www.w3.org/TR/xhtml-media-types/
- http://www.fileformat.info/info/mimetype/index.htm
- http://interval.cz/vyvoj-aplikaci/net/
- http://interval.cz/clanky/download-se-spravnym-content-type-v-asp-net/
- http://interval.cz/clanky/grafy-v-asp-net-kolace/
- http://www.zvon.org/xxl/XPathTutorial/General_cze/examples.html
- http://www.kosek.cz/xml/xslt/vyrazy.html

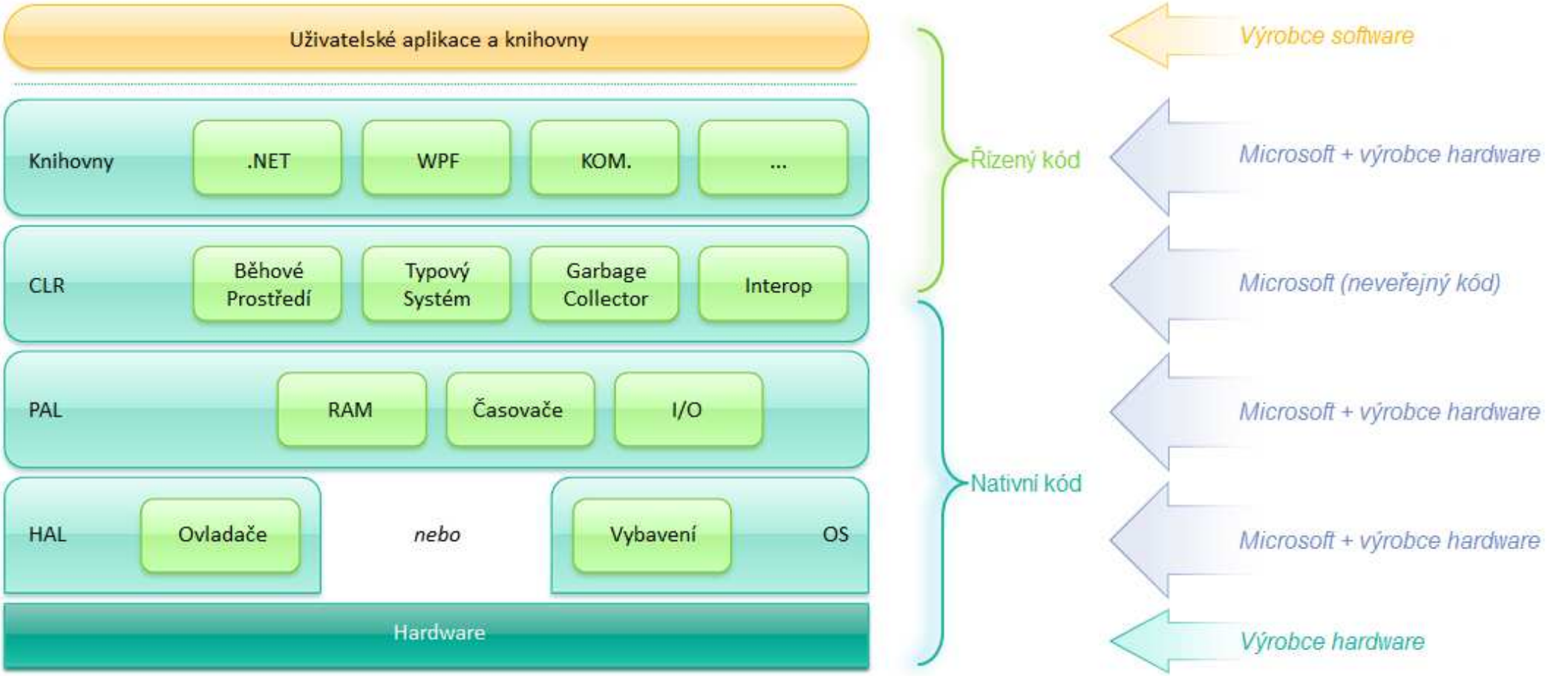
MS .NET Micro Framework

- „Malé“ běhové prostředí .NET pro nejmenší aplikace bez OS
  - Součástí FW je také podpora debug nativně v C#
- Programování v C#, od v4.2 i VB.NET
- Dostupné zdrojové kódy od v4 – codeplex.com
  - aktuální verze 4.3 (4.4), zpětně zahrnuje i starší 4.x
  - GUI založeno na WPF = definice vzhledu v XAML souboru
- Požadavky
  - 32-bitový procesor (typicky ARM), pro který je hotový port nebo si jej vytvořit pomocí **porting-kit**
  - minimálně 256kB Flash a 64kB RAM
  - Visual Studio (pro 4.x nutné min. VS2010)
- Přeneseny některé jmenné prostory (cca 70 tříd), např.
  - Sockets, vyžadují více paměti
  - Threads
- Doplněn Microsoft.SPOT
  - HW záležitosti – Digi In, Out, Analog, PWM, …
- Ostatní HW specifické jmenné prostory dodává výrobce hw

Micro.NET FW – platformy

- Pomocí Porting-kit k dispozici FW pro SMT32 (Cortex-M4) – např. výkonější STM Discovery kity
  - Existuje např. port pro Nucleo STM32F411
- Hlavní výrobce – **GHI Electronics**
  - USBizi – počáteční kompatibilita s Arduino
    - NXP LPC2387 - ARM7, 72MHz, 512kB flash (148k user), 96kB RAM (62k user), USB, CAN, SPI, …
    - FEZ Panda II + Connect Shield (WizNET 5100 Ethernet řadič)
  - G80 (G120) – SystemOnModule
    - STM32F4xx, 180MHz, 192kB (2MB) Flash, 152kB (13MB) RAM
  - G400 – SystemOnModule
    - ARM9, 400MHz, 4MB Flash, 128MB DDR
    - podpora periférií typu LCD, CAN, WiFi, …
    - řada desek s různým přidaným HW a konektivitou
- Netduino – STM32F4(2), Arduino pinout
  - 168MHz, 384kB flash, 100+kB RAM, Ethernet (verze Plus)
- Další moduly především ve formě SOM
  - bývalý uCSimply - dostupný port micro.net i Linux s komplet dokumentací v CZ
    - ARM9, 180MHz, 32MB SDRAM, 256MB Flash, Ethernet

Blokově části .NET Micro FW



Příklad programu v .Net Micro FW 4.1

```
using System;
using System.Threading;
using Microsoft.SPOT;
using Microsoft.SPOT.Hardware;
using GHIElectronics.NETMF.FEZ;
using GHIElectronics.NETMF.Hardware;

namespace PandaApp
{
  public class PandaAppProgram
  {
    public static void threadAnalog()
    {
      AnalogIn pot = new AnalogIn((AnalogIn.Pin)FEZ_Pin.AnalogIn.An2);
      pot.SetLinearScale(0, 100);

      while (true)
      {
        int x = pot.Read();
        Debug.Write(x.ToString);

        Thread.Sleep(100);
      }
    }

    public static void Main()
    {
      Thread thAnalog = new Thread(threadAnalog);
      thAnalog.Start();
      // Blink board LED

      OutputPort ledBase = new OutputPort((Cpu.Pin)FEZ_Pin.Digital.LED, ledState);

      while (true)
      {
        Thread.Sleep(500);

        ledBase.Write(!ledBase.Read());
      }
    }
  }
}
```

TinyCLR OS a STM32F411RE

- Poslední komunitní verze Micro.NET FW je 4.4 včetně příkladů portace na vlastní HW
- Firma GHI Electronics rozvinula Micro.NET do projektu TinyCLR OS
  - Kompletně přepsáno jádro = nové třídy a funkcionalita
  - Nové možnosti portování
  - K dispozici jako GitHub projekt
    - Překlad z C++ platformně závislých kódů pomocí Keil MDK nebo ARM-GCC
    - Možno upravit pro vlastní platformu
  - Podpora pluginy pouze od VS2017 výše (včetně Community edition)
- Zatím verze 1.0.0 preview 2 (říjen 2018)
  - Některá funkcionalita je stále "plánována" – např. networking
  - Informace a dokumentace v trvalém vývoji (fórum)
  - Ověřená podpora pouze malého množství HW, především od výrobce GHI
  - Řadu vhodných objektů je třeba si zatím "doprogramovat ručně"
  - Od v1.x plugin pro VS2017 v "extensions" a knihovny v NuGet
- V repozitáři podpora pro desky USBizi (včetně Panda II), ale je do verze 0.12.0
- Hlavně připraven port pro Nucleo STM32F411RE
  - Používáme na výuku KAE/MPP a KAE/MINA
  - Pro otestování funkce stejný shield jako v KAE/MPP
  - Lze upravit konfigurace pro využití "našich" periférií na jiných než defaultních portech (PWM, UART, ...) – připraveno v EL510
  - Nutné debug připojení přes USB
    - 32F411 má USB-device funkcionalitu na pinech PA11 a PA12
    - Vhodné připojit přes mini- nebo micro-USB kabel
    - Napájení přes USB ST-Link zůstává, lze využít UART
      - Slouží k nahrání booloaderu příp. kompletního firmware

Příklad TinyCLR programu pro STM32F411RE a MPP shield

```
using GHIElectronics.TinyCLR.Devices.Gpio;
using GHIElectronics.TinyCLR.Devices.Pwm;
using GHIElectronics.TinyCLR.Devices.Spi;
using GHIElectronics.TinyCLR.Pins;
using System;
using System.Collections;
using System.Diagnostics;
using System.Threading;

namespace MPP_deska
{
  public static class ExtMethods
  {
    public static void Write(this GpioPin pin, bool state)
    {
      pin.Write(state ? GpioPinValue.High : GpioPinValue.Low);
    }

    public static GpioPin OpenPin(this GpioController ctrl,
      char port, byte pin,
      GpioPinDriveMode mode = GpioPinDriveMode.Output)
    {
      GpioPin newPin = ctrl.OpenPin(PinNumber(port, pin));
      newPin.SetDriveMode(mode);
      return newPin;
    }

    public static int PinNumber(char port, byte pin)
    {
      if (port < 'A' || port > 'E')
        throw new ArgumentException();

      return ((port - 'A') * 16) + pin;
    }
  }
  ...
}
```

```
...
class Program
{
  static void Main()
  {
    Debug.WriteLine("MPP deska ");

    GpioController ctrl = GpioController.GetDefault();

    GpioPin ledOE = ctrl.OpenPin('A', 9);
    GpioPin ledLE = ctrl.OpenPin('A', 8);

    SpiDevice spiLed = SpiDevice.FromId(
      GHIElectronics.TinyCLR.Pins.FEZ.SpiBus.Spi1,
      new SpiConnectionSettings(ExtMethods.PinNumber('C', 10))
      {
        ClockFrequency = 10 * 1000 * 1000,
        DataBitLength = 8,
        Mode = SpiMode.Mode0  //
      });

    ledOE.Write(false);       // enable outputs

    byte b = 1; // 0;
    while (true)
    {
      spiLed.Write(new byte[] { b });
      ledLE.Write(true);
      ledLE.Write(false);

      Thread.Sleep(50);
      b++;
    }
  }
}
```

NanoFramework

- Poměrně "čerstvý" projekt bez zázemí konkrétní HW firmy
  - Intenzivně vyvíjený
  - Řada vzorových HW implementací pro STM32x + komunitní řešení
  - Existuje port pro ESP32
  - Viditelná inspirace původním Micro.NET FW
  - Funkčnost debuggeru součástí FW
  - Díky základu ChibiOS (RT OS) a LwIP (knihovna pro TCP/IP komunikaci) má poměrně slibný potenciál
- Podle dokumentace snadný interop s nativním kódem
  - "knihovna" funkcí v C/C++
  - Definovány základní typy pro možnost předvání parametrů a výsledků
- Oproti TinyCLR komplikovanější proces tvorby "portu"
  - Vyžaduje více nástrojů, knihoven, ...

NanoFramework – blinky

```
using Windows.Devices.Gpio;
using System;
using System.Threading;

namespace Blinky
{
  public class Program
  {
    public static void Main()
    {
      // PA5 is LED_GREEN in Nucleo64 STM32Fxx
      GpioPin led = GpioController.GetDefault().OpenPin(PinNumber('A', 5));

      led.SetDriveMode(GpioPinDriveMode.Output);

      while (true)
      {
        led.Toggle();
        Thread.Sleep(500);

        Console.WriteLine("Hello from nanoFramework!");
      }
    }

    static int PinNumber(char port, byte pin)
    {
      if (port < 'A' || port > 'J')
        throw new ArgumentException();

      return ((port - 'A') * 16) + pin;
    }
  }
}
```

Zajímavé odkazy

- http://informatix.miloush.net/microframework/Tutorials/WhatIs.aspx
- http://www.microsoft.com/en-us/netmf/default.aspx
- http://en.wikipedia.org/wiki/.NET_Micro_Framework
- http://en.wikipedia.org/wiki/.NET_Framework
- http://www.hanselman.com/blog/TheNETMicroFrameworkHardwareForSoftwarePeople.aspx
- http://www.netmf.com/Home.aspx
- http://www.tinyclr.com/
- https://www.ghielectronics.com/catalog/product/256
- https://www.ghielectronics.com/catalog/product/261
- http://shop.microframework.eu/
- http://netduino.com/ resp. https://www.wildernesslabs.co/
- http://www.ucsimply.cz/products/modsam9260/
- http://blogs.msdn.com/b/laurelle/archive/2011/10/06/creating-and-launching-timer-in-net-microframework.aspx
- https://nanoframework.net/
- https://github.com/nanoframework/Samples/blob/master/Blinky/Program.cs

- https://os.mbed.com/platforms/ST-Nucleo-F411RE/
- http://docs.ghielectronics.com/tinyclr/boards/stm32_boards.html

Přednáška 6 - Datové přenosy

- Zabezpečení bloků dat prakticky
  - jednoduché součty
  - CRC
- Zabezpečení přenosu dat
  - na úrovni protokolu - "automaticky"
  - vlastním kódem
- Šifrování a hash (heše)
  - Hash
  - Symetrická a asymetrická šifra
- Třída SerialPort v rámci .Net

- Bezdrátové technologie pro přenos dat (lokální)

Zabezpečení bloků dat

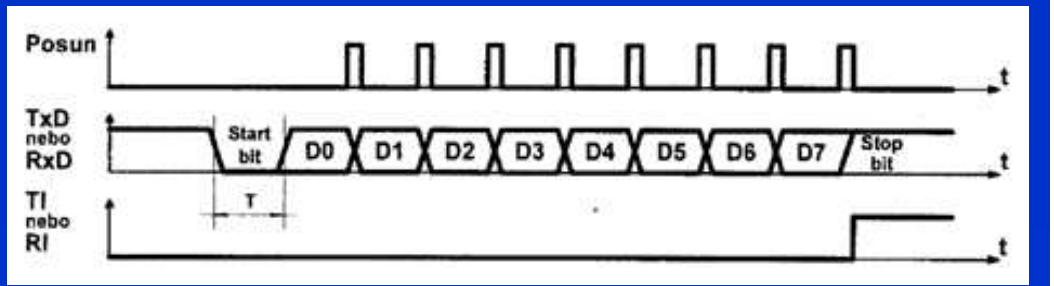
- Přijímací strana si musí být "jistá", že data jsou v pořádku
- Přidává se kontrolní informace
  - parita
    - pro méně náročné přenosy
    - zachytí pouze chybu 1 bitu
    - typicky HW detekce
  - kontrolní součet
    - zachytí více-bitové chyby
    - výpočetně náročnější

Zabezpečení jednoduchými součty

- Nejjednodušší na implementaci
- Typicky se provede 8-bitový součet bez přenosu nebo XOR přes všechna data
- Za data se připojí 1 byte obsahující
  - hodnotu součtu - přijímač ji pak porovná
  - doplněk - přijímač vypočte kontrolní součet přes všechna data + byte doplňku a výsledek musí být 0
- Typicky se neodhalí vícebitové chyby

Zabezpečení paritou

- Parita se používá při sériovém přenosu
  - **None** = žádná
  - **Mark** = značka - paritní bit vždy 1
  - **Space** = mezera - paritní bit vždy 0
  - **Even** = sudá - paritní bit doplňuje na sudý počet 1
  - **Odd** = lichá - paritní bit doplňuje na lichý počet 1
- Paritní bit se přidává při odesílání
- Pokud při příjmu nesouhlasí parita, je nastaven v řadiči příznak "**Parity Error**"
- Neplést s "Frame Error" - neodpovídají pozice start-bitu nebo stop-bitu nebo délka stop-bitu (má-li být > 1)



CRC - Cyclic redundancy check

- Speciální typ hashovací funkce
- Vypočteno přijímačem i vysílačem - shoda !!
- Výsledkem je zbytek po dvojkovém dělení
  - bez přenášení bitů (operace XOR)
  - dělí se vstupní data binární posloupností odpovídající koeficientům polynomu
- Ne každý polynom generuje CRC
  - standardizováno několik vybraných
- CRC neumí detekovat nadbytečné nuly před a za užitečnými daty
- Algoritmické řešení
  - tabulka - spotřebuje data, rychlá
  - přímý výpočet - pomalejší, paměťově úspornější

CRC16 - prakticky v C/C++

```
#define word unsigned int
#define byte unsigned char

word CRC16(byte *MsgBuffer, word DataLen)
{
  word CRC = 0xFFFF;
  byte a;
  byte LSB;

  while (DataLen--)
  {
    CRC ^= (word)(*MsgBuffer++);
    for (a = 0; a < 8; a++)
    {
      LSB = CRC & 0x0001;
      CRC >>= 1;

      if (LSB == 0x0001)
      {
        CRC ^= 0xA001;
      }
    }
  }
  return CRC;
}
```

CRC32 - prakticky v C#

```
/// x^32+x^26+x^23+x^22+x^16+x^12+x^11+x^10+x^8+x^7+x^5+x^4+x^2+x+1.
internal sealed class Crc32 : IChecksum
{
  readonly static uint CrcSeed = 0xFFFFFFFF;
  readonly static uint[] CrcTable = new uint[] {
    0x00000000, 0x77073096, 0xEE0E612C, 0x990951BA, 0x076DC419,
    ... };

  internal static uint ComputeCrc32(uint oldCrc, byte bval)
  {
    return (uint)(Crc32.CrcTable[(oldCrc ^ bval) & 0xFF] ^ (oldCrc >> 8));
  }

  uint crc = 0;

  public void Update(byte[] buffe
  {
    Update(buffer, 0, buffer.Lengt
  }

  public long Value
  {
    get { return (long)crc; }
    set { crc = (uint)value; }
  }

  ...
}
```

```
public void Update(byte[] buf, int off, int len)
{
  if (buf == null)
    throw new ArgumentNullException("buf");

  if (off < 0 || len < 0 || off + len > buf.Length)
    throw new ArgumentOutOfRangeException();

  crc ^= CrcSeed;

  while (--len >= 0)
    crc = CrcTable[(crc ^ buf[off++]) & 0xFF] ^ (crc >> 8);

  crc ^= CrcSeed;
}
```

Zabezpečení přenosu - požadavky

- Maximální spolehlivost
  - detekována platnost přijatých dat
  - pro kritická data přidány kryptografické výpočty
- Opravitelnost chyb
  - "samoopravitelnost" kódu prodlužuje datový blok
- Minimální režie
  - nutno zvolit optimální řešení podle nároků aplikace a možností přenosové cesty

Ověření přenosu dat - handshake

- **HS** - komunikace přijímač-vysílač o stavu přenosu
- Prakticky všechny průmyslové protokoly jsou založené na "odpovědi přijímače"
- Odpověď na každý blok (=potvrzení)
  - příliš velká režie množstvím dat odpovědi
  - časová režie na timeout nebo nutnost "rychlé odpovědi"
  - jednoduchá implementace
- Potvrzení více bloků
  - vysílač nečeká po každém bloku
  - podle potvrzení si "odškrtne" odeslané A potvrzené
  - nejvýhodnější je umět odesílat bloky mimo pořadí
- V odpovědi může být i optimalizace velikosti bloku dat
  - ideálně pak adaptivní tak, aby pokryla dynamické parametry linky

TCP protokol

- RFC793 - "spojově orientovaný protokol pro přenos toku bajtů na transportní vrstvě se spolehlivým doručováním"
- Na rozdíl od UDP se jedná o datový kanál, tedy přijímač i vysílač musí být aktivní
- Při přenosu se data rozdělí na pakety, které mohou putovat různými cestami
- Přijímač potvrdí příjem a složí pakety dle pořadí do výsledku
- Vysílač používá vysílací okno, kdy odesílá pakety dříve, než dostane potvrzení
- Přijímač může potvrdit příjem "balíku" paketů najednou
- TCP je poměrně komplikovaný protokol

SerialPort - inicializace

- Konstruktor typicky obsahuje číslo portu ve formě řetězce - COM1 - COMx (dle systému)
  - pro neplatné COM dává IOException
  - pro COM>9 se používá "\\\\.\\COM10" (C-zápis \\)
  - možno předat i rychlost, paritu a počet bitů
  - enumerace portů v systému:

```
foreach (String s in System.IO.Ports.SerialPort.GetPortNames())
{ ... }
```

- Vlastnosti objektu
  - nastavení portu - rychlost, parita, ...
  - stavy řídících signálů portu
  - nastavení zpracování dat

SerialPort - přenos dat

- Funkce Open() a Close()
- Čtení dat
  - Read - do pole
  - ReadByte - čte jeden BYTE z bufferu SP
  - ReadChar - čte jeden ZNAK
    - dle nastaveného Encoding !!
- Události - pozor, SP má vlastní Thread
  - DataReceived - "něco přišlo" = zpracovat přijatá data podle počtu BytesToRead
  - ErrorReceived - typicky parity- nebo frame-error
  - PinChanged - bity CD, CTS, DSR, RI

SerialPort - HandShake

- Nejjednodušší možnost - žádný HS
  - přijímač musí s rezervou stihnout zpracovat KAŽDÝ bajt zprávy (příp. se využívá přijímací buffer)
- RTS/CTS - využití řídících bitů portu
  - přijímač nastavuje RTS signál do neaktivního stavu pokud se mu zaplnil přijímací buffer, uvolňuje do aktivního stavu když data zpracoval
- XON/XOFF - řídící slova v toku dat
  - přijímač posílá vysílači hodnotu XOFF (0x13), aby přestal vysílat, když se mu zaplnil přijímací buffer (typicky z 75%)
  - když se buffer vyprázdnil, posílá XON (0x11), že je možné opět vysílat

SerialPort příklad

```
int _readPtr = 0; int[] _rxbuf = new int[BUFFERLEN];

void port_DataReceived(object sender, SerialDataReceivedEventArgs e)
{
  if (e.EventType != SerialData.Chars)    // nebyly to data ?
    return;

  while (((SerialPort)sender).BytesToRead > 0)  // dokud je co cist
  {
     _rxbuf[_readPtr] = ((SerialPort)sender).ReadByte();  // uloz do bufferu
     if (!_processing)
       continue;
     ...                 // zpracovani prijatych dat
  }
}

public SendData(byte bReg)
{
  _processing = true;
  ...                    // priprava bloku dat k odeslani
  _readPtr = 0;
  _timeoutCnt = 2 * (int)(1000 / _tm.Interval);      // interval je v ms
  _port.Write(buf, 0, 4);
}

int _timeoutCnt = 0; bool _processing = false;

void tm_Elapsed(object sender, System.Timers.ElapsedEventArgs e)
{
  if (_timeoutCnt > 0)
  {
    _timeoutCnt--;
    if (_timeoutCnt == 0)
    {
      if (_processing)
      {
         _processing = false;
         ...                          // reaguj na timeout pri prijmu odpovedi
      }
    }
  }
}
```

Šifry (kryptografie)

- Blokové
  - Vstupní data zpracovávají po blocích definované délky
- Proudové
  - Vstupní data zpracovávají pro bitech

- Symetrické
  - Pro šifrování i dešifrování stejný klíč
  - AES, DES, ...
- Asymetrické
  - Privátní a veřejný klíč
  - Generované klíče založené na "velkých" prvočíslech příp. eliptických křivkách apod.
  - RSA, D-H (Diffie-Hellman)

- Symetrické šifry se často používají společně s asymetrickými:
  - Vstupní data se zašifruje symetrickou šifrou s náhodně vygenerovaným klíčem
  - Symetrický klíč se zašifruje veřejným klíčem asymetrické šifry
  - Dešifrovat data může pouze majitel tajného klíče dané asymetrické šifry

HASH

- = otisk (fingerprint)
  - Nesmí být moýnost zjistit obsah zprávy z hashe
- Např. pro ukládání hesel
  - Připojen ještě "salt" pro prodloužení bloku dat
  - Při přilogování se porovnává hash hesla s hashem v DB
  - Při úniku dat z DB nejsou hesla čitelná
- CRC
- MD5 – jednoduché, ale zastaralé (= málo bezpečné)
- SHA2 s různým počtem bitů

Šifry a hashe - implementace

- C# - jmenný prostor System.Security.Cryptography
- Knihovny a utility
  - OpenSSL (často v Unix repozitářích)
  - BouncyCastle (port z Java knihovny)
  - + Řada komerčních řešení

- Přímo v HW na mikrokontroléru
  - Cryptographic processor (CRYP) u vyšších řad
    - STM32F415/7 a F43x
    - DES, 3DES, AES
  - CRC blok – typicky u Ethernetu – STM32F4xx (např. i F411)
    - CRC-32 (Ethernet) polynomial = 0x4C11DB7
      - X32 + X26 + X23 + X22 + X16 + X12 + X11 + X10 +X8 + X7 + X5 + X4 + X2+ X +1
    - Single input/output 32-bit data register
    - "CRC computation done in 4 AHB clock cycles (HCLK)"

Zajímavé odkazy

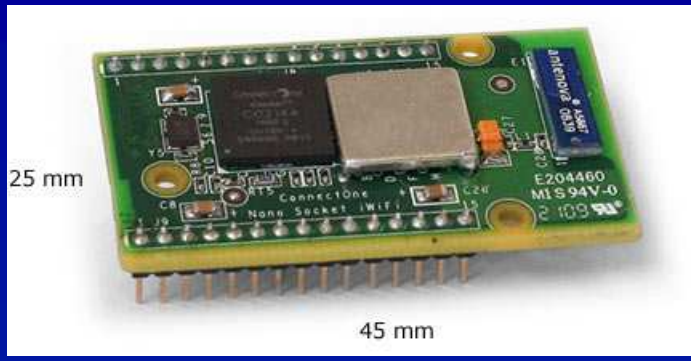
- http://cs.wikipedia.org/wiki/CRC
- http://www.geocities.com/SiliconValley/Pines/8659/crc.htm
- http://cs.wikipedia.org/wiki/S%C3%A9riov%C3%BD_port
- http://cs.wikipedia.org/wiki/TCP
- http://msdn2.microsoft.com/en-us/library/system.io.ports.serialport_members.aspx

- https://cs.wikipedia.org/wiki/Ha%C5%A1ovac%C3%AD_funkce
- https://cs.wikipedia.org/wiki/Message-Digest_algorithm
- https://cs.wikipedia.org/wiki/Asymetrick%C3%A1_kryptografie
- https://cs.wikipedia.org/wiki/Symetrick%C3%A1_%C5%A1ifra
- http://www.codeproject.com/Articles/769741/Csharp-AES-bits-Encryption-Library-with-Salt
- http://www.bouncycastle.org/

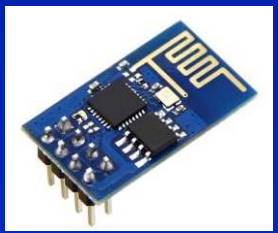
Bezdrátové technologie

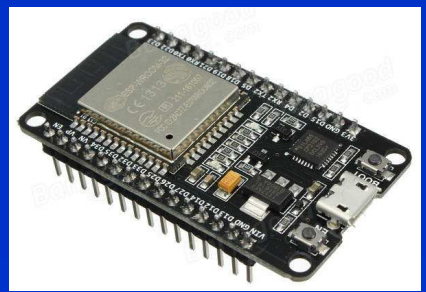
- WiFi
- BlueTooth
- ZigBee
- Bezlicenční pásmo pro data
  - 433MHz
  - 868HMz
- Speciální technologie

WiFi v elektronice



- Nejjednodušší připojení v případě řídící elektroniky založené na PC a moderních OS
- Přenosové pásmo 2.4GHz (802.11b/g sítě) nebo 5GHz (802.11a), aktuální 802.11n obě
- Vlastnosti
  - velká teoretická propustnost dat
    - maximálně 11 (b), 54 (g), 600(n)MBit
  - standardní protokoly
  - jednoduché připojení k ethernetové síti
  - režim provozu Client i AP (=server pro stanice)
  - pásmo 2.4GHz často využívané - rušení !
- Moduly připojené SPI nebo UART, často obsahují přímo mikrokontrolér (ARM) + GPIO a sběrnice = není třeba další HW
  - Espressif ESP8266, ESP32
  - ST - SPWF04Sx (NRND ☹)
  - Microchip, ...





BlueTooth

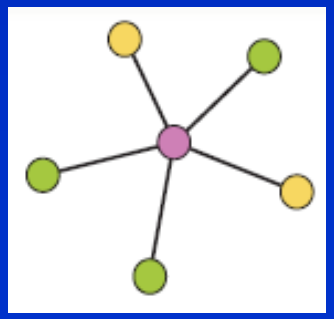
- Pásmo 2.4GHz, přepínání kanálů (FHSS)
- Určeno pro trvalé přenosy malých dat
- Vlastnosti
  - relativně malá propustnost dat (kolem 720kbit/s verze 1.2)
  - EDR (2.1) – 1.4Mb/s
  - rychlejší mód HS (3.0) – navázání spojení a následně přenos WiFi
  - dosah 10m typicky (Class 2)
  - vytvořena síť komunikujících buněk
- BLE = BT Low Energy (aka BT4)
  - Nové profily – heart rate, medical, battery, ...
  - "Master" vyjedná s klientem nabízené služby/profily (GATT transakce)
  - Pro Windows podpora až od Win8, lépe Win10
  - Přímá podpora v Android a iOS = aplikace nejsou pro PC ☹

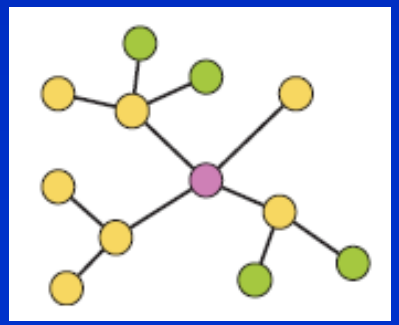
Bluetooth (2, 3) podporuje jak dvoubodovou, tak mnohabodovou komunikaci.

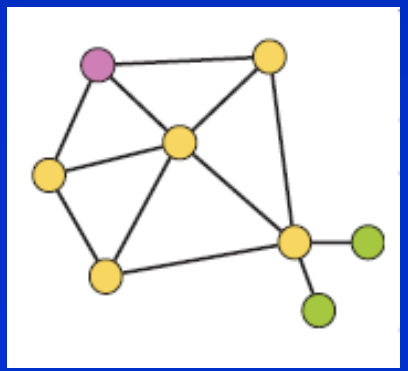
Pokud je více stanic propojeno do ad hoc sítě, tzv. pikosítě (piconet), jedna rádiová stanice působí jako řídící (master) a může simultánně obsloužit až 7 podřízených (slave) zařízení. Všechna zařízení v pikosíti se synchronizují s taktem řídící stanice a se způsobem přeskakování mezi kmitočty. Specifikace dovoluje simultánně použít až 10 pikosíti na ploše o průměru 10 metrů a tyto pikosítě dále sdružovat do tzv. „scatternets“ neboli „rozprostřených“ sítí.

ZigBee

- Pásmo 2.4GHz (alternativně 868/915MHz)
- Standardizováno v rámci IEEE802.15.4
- Určeno pro přenosy dat ze senzorů s důrazem na nízkou spotřebu
- Různé struktury sítě uzlů
- HW možno použít i pro méně sofistikované protokoly
  - Freescale
- Komplet Stack - cca 10k$, jinak např. profil sériového portu
- Často používaný HW XBee
  - různé varianty podle výkonu/dosahu a vybavení FW







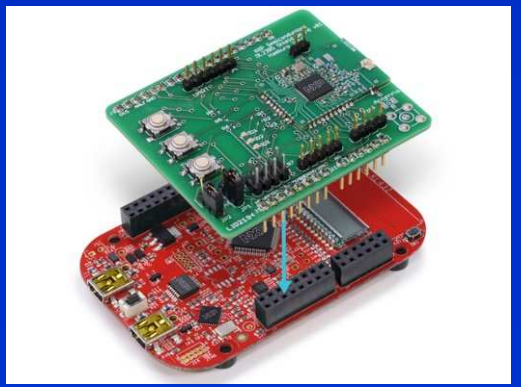
- **ZigBee Coordinator (ZC)** establishes and controls the ZigBee network
- **ZigBee Router (ZR)** can support data routing functionality
- **ZigBee End Device (ZED)** cannot support data routing functionality

Datové přenosy v pásmu 433MHz

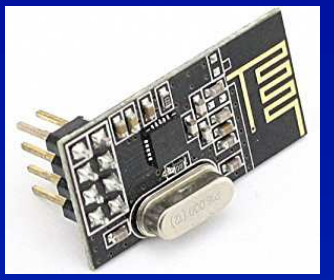
- Typicky Half-Duplex (přepínání směru)
  - přepnutí má časovou režii (cca 100ms až <500ms)
  - AM nebo FM modulace
  - přenosy typicky jednotky kbit/s
- Používáme vícekanálové moduly Aurel s přenosem max. 38.4kbit/s

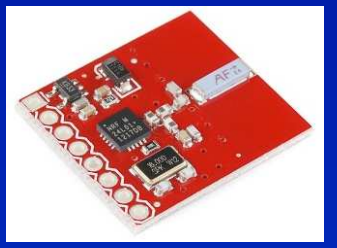
Speciální přenosy

- Pásmo 2.4GHz
  - ve sportovní/lékařské oblasti např. protokol ANT/ANT+
  - Např. obvod NRF24AP2
- Čipy pro SubGig přenosy nabízejí řada výrobců, namátkou:
  - ST – RF transceiver SPIRIT1, konfigurovatelný data-rate 1-500kb/s
    - Modul SPSGRF součástí řady kitů a Shieldů
  - Freescale (nyní NXP) – 1.2-300kb/s
  - Nordic, TI, ...
  - pásma 315, **433**, 470, **868**, 915, 928, 960MHz



Speciální přenosy





- Nordic moduly
  - Např. nRF24L01
    - Pásmo 2.4GHz
    - Přenos až 2Mbit
    - UltraLow Power, minimum externích součástek
    - Vlastní paketový protokol
  - SubGig moduly
    - Pásmo 433/868/902-928MHz
- Nordic jednočipové mikropočítače
  - Jádro x51
    - 433/868MHz (SubGigaHertz) nebo 2.4GHz
    - ShockBurst mode
      - komprese datového paketu na přenos max. rychlostí
      - menší množství poruch díky kratšímu času
    - až 1Mbit/s
  - Jádro ARM
    - Založené na Cortex-M0+ a M4F
    - Komunikace v pásmu 2.4GHz – BlueTooth LE, ANT+ a nRF protokol

Zajímavé odkazy

- http://cs.wikipedia.org/wiki/Bluetooth
- http://en.wikipedia.org/wiki/Bluetooth
- https://learn.adafruit.com/introduction-to-bluetooth-low-energy/gatt
- http://en.wikipedia.org/wiki/ZigBee
- http://www.hw.cz/embedded/xbeexbee-pro-moduly-digi.html
- http://www.zigbee.org/en/certification/compliant_platforms.asp
- http://www.silabs.com/tgwWebApp/public/web_content/products/Microcontrollers/en/ZigBee.htm
- http://www.nvlsi.no/index.cfm?obj=menu&act=displayMenu&men=21
- http://www.czechlabs.cz/pages/cenik.php?lang=cz&cap=-&categ=aurel
- http://www.nordicsemi.com/eng/Products/2.4GHz-RF
- http://en.wikipedia.org/wiki/ANT_(network)
- http://www.st.com/en/wireless-connectivity/sub-1ghz-rf.html?querycriteria=productId=SC1845
- https://www.nxp.com/products/rf/low-power-tx-rx-ics/sub-ghz-rf:MC_71594

Přednáška 7 - různé - I

- Reverse Engineering (nejen .NET)

- PDA (Windows i Palm)
- Bezdrátové technologie pro aplikace

- OS a požadavky/vazby na HW
- Embedded Linux
  - Vývojový kit Galileo

- RFID
- GPS
- Datové mobilní komunikace

Reverse Engineering

- ROTOR - zdrojové kódy implementace základu .NET Frameworku a C#
- .NET mezikód (IL) lze převést zpět na kód v C# nebo VB
  - Reflector - prohlížeč tříd + disasembler
  - Anakrino - vygeneruje kód
- Dotfuscator
  - provádí "obfuskaci" (zmatení) kódu
  - přejmenování proměnných, metod a tříd
  - přehodí instrukce - zachování funkčnosti, ale neodpovídají konstrukci z např. C#

- Debugger
  - např. SoftICE
  - umožňuje sledovat běh aplikace, krokovat, ...

Mobilní klienti – začalo to PDA

- Hodně průmyslově využívaných zařízení stále založeno na mobilních Windows v různých verzích
- Nové systémy založeny buď na obdobě PC (Windows/Linux) nebo OS pro tablety/mobily (Android a spol.)

- Starší alternativou PalmOS
  - prakticky pouze HW firmy PALM
  - zavedena ploška Grafitti - psaní znaků
  - konektivita IrDA, WiFi, BT
  - HW kompatibilita založena na M68k architektuře
  - poslední HW na platformě Intel PXA
  - pseudo-multitasking založený na úplném přepínání aplikací (musí si mezi-uložit stav)
  - v poslední verzi možnost rezidentních aplikací na pozadí (omezen počet)
  - programování C, Java, existovalo patřičné SDK i emulátor

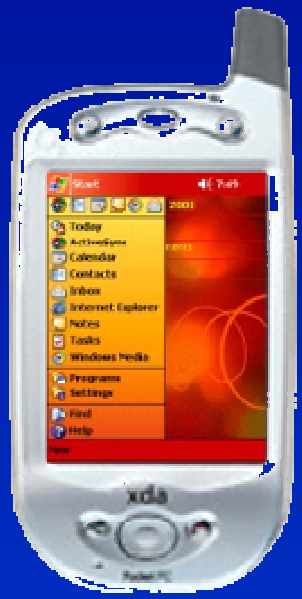
Mobilní klienti - současnost

- Bezdrátová konektivita
  - WiFi (b/g)
  - BlueTooth – přenos dat i audio (nejen HF)
  - GPRS, EDGE, sítě 3.5G (UMTS, HSxPA, …), příp. LTE
- Dotykové ovládání
  - včetně vícedotykového
- Možnost zpracování Office dokumentů
- GPS modul s mapovými aplikacemi
- Akcelerometry, kompas
- Předpokládá se trvalá konektivita zařízení
  - např. e-mailové aplikace, sociální sítě

PDA terminologie Microsoftu ...

Windows SmartPhone

Pocket PC



Pocket PC Phone Edition

Windows Phone 7, 7.5, 8, 8.1

... a jak to dopadlo ?

Mobilní Windows

- Několik vývojových verzí
  - PocketPC 2002 (WinCE 3.x)
  - Windows Mobile 2003 (založeno na WinCE 4.20)
  - Windows Mobile 2003SE (režim "portrait") = WinCE 4.21
  - stále aktuální - Windows Mobile 5.0 (WinCE 5.0)
  - běžné - Windows Mobile 6.0, 6.1, 6.5 (2009) – poslední „použitelné"
  - Windows Phone 7 (novinka 2010) – prakticky nulová kompatibilita
  - Windows Phone 8 (novinka 2012) – opět pouze pro telefony
  - Windows 10 Phone – (2017) zřejmě poslední ☹
- Typické vlastnosti před WinPhone
  - rozlišení obrazovky - QVGA (240x320), VGA (480x640)
  - SW původně založen na Hitachi SH3, MIPS, ARM
  - později výhradně Intel PXA (ARM9 jádro), resp. kompatibilní ARM (Samsung)
  - součástí systému Pocket Office, Media Player a Pocket IE
  - podpora BT, Wifi, IrDA, USB Host (díky jádrům PXA270 a lepším)
  - aplikace mohou využít Compact .NET Framework či Compact SQL databáze
  - podpora Visual Studia včetně emulátoru (poslední VS2008 s .NET 3.5)
  - výrobci si upravovali ovládání ve formě nadstaveb (dotyky apod.)

Ostatní mobilní platformy

- Symbian
  - podpora – nyní opět pouze Nokia (pokusy Samsung, SE)
  - Open Source (od 2009), od 2010 pod správou Nokia
  - programování Carbide.C++, alternativně Java, .NET
- Palm WebOS
  - Palm Pre (zřejmě jediná platforma od 2009)
  - založené na Linuxu
  - koupeno HP, kód uvolněn, prodej skončil
- iOS (Apple)
  - vývoj pouze pod Mac OS X
  - nativní programování v Objective-C a Cocoa, nově jazyk Swift
  - masivní podpora multidotykového ovládání
  - Xamarin – C#, nutný Mac pro Build/Deploy (v síti)
- Google Android
  - programováno v Javě, vlastní běhové prostředí Dalvik
  - jádro v principu založené na Linuxu
  - zázemí Google – hlavní mobilní přístup ke službám G.
  - nasazován také na embedded počítače
  - Xamarin (dříve MonoDroid) – C#, možnost společných knihoven v .Net
- Několik zařízení postaveno na „čistém" Linuxu

Operační systémy a HW

- Zjednodušení - OS je prostředí pro spouštění a řízení procesů
- RTOS pro mikropočítače (i pro 8-bity)
  - přidělování procesoru úlohám
  - typicky kompilován spolu s aplikací
  - jednotlivé procesy tvořeny typicky funkcí s vlastní while(1) smyčkou a čekáním na nějaké události
- "Normální OS" = práce s "aplikacemi"
  - typicky podporuje i vícevláknovost
  - aplikace musí běžet z libovolné adresy
  - pro přepínání aplikací typicky nutný memory-management založený na stránkování paměti (počínaje i386)
  - ochrana paměti procesů je vhodná také na HW úrovni
  - abstraktní vrstva nad HW, aplikace musí přistupovat prostřednictvím OS

OS pro řídící systémy

- PC kompatibilní HW
  - Windows CE
  - Windows XP Embedded, Win Embedded 7
    - Target Designer – vytvoří OS na míru
    - profily Thin Client, NavReady, POSReady (Point of Service), …
  - Windows 10 IoT
    - Běží na ARM v7+ (např. Raspberry PI2, 3), multitask, single user
    - Programování v JS+HTML, C#/C++ + XAML, UWP aplikace
  - Windows 10 IoT Enterprise
    - Obdoba Win embedded = možno přizpůsobit jedinému HW
  - Linux Embedded
  - Android port (http://www.android-x86.org/)
- Vlastní HW – v současnosti nejspíš na ARM9/Cortex-A/Cortex-M
  - Nutný BSP (Board Support Package) – úpravy jádra OS + ovladače pro HW
  - WinCE (Hitachi, ARM, ...) – poslední v6
  - Windows Embedded Compact (založené na W7)
  - Android – někdy jen 1.6 nebo 2.x, u nových 4.x
  - Linux – nutno kompilovat pro cílovou platformu
  - µC Linux (např. na FPGA 32-bitových jádrech, ale i MC68000)
- RTOS
  - QNX
  - Rozšíření Windows (náhrada knihoven jádra) - Pharlap

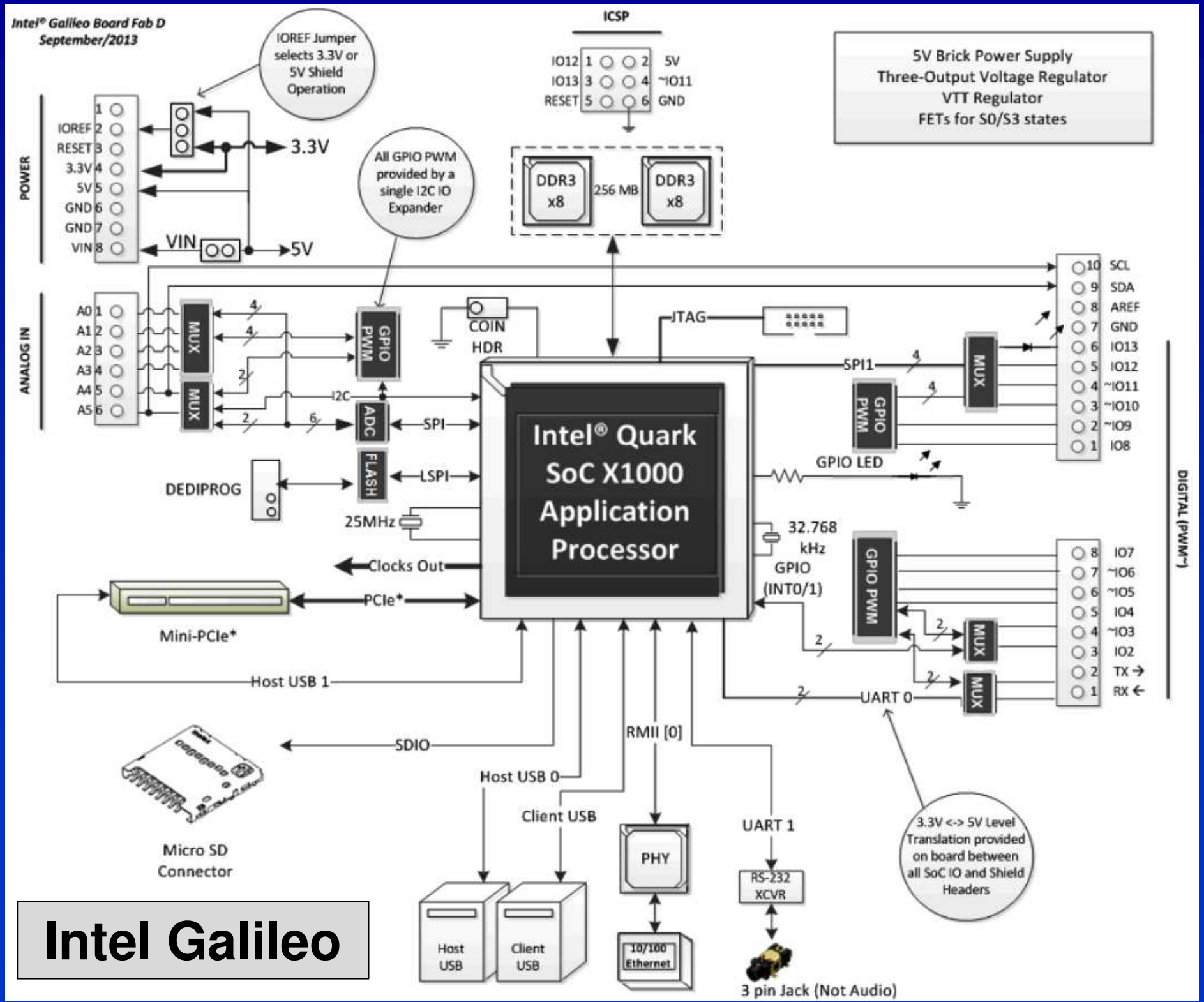
Embedded Linux

- Varianty s grafickým prostředím (HW má HDMI/VGA/... výstup)
- Varianty "bez obrazovky" – nutno se připojit Terminálem z jiného počítače
  - Pomocí sériové linky (typicky ukazuje už "start" systému, parametry v konfiguraci zavaděče)
  - Pomocí SSH (výjimečně jen Telnet) – většinou k dispozici Ethernet, základní konfigurace by měla podporovat DHCP pro získání IP adresy
- Různé distribuce podporované výrobcem HW nebo komunitou
  - Založené na **Debian**
    - Balíčkovací systém založený na dpkg = instalace pomocí apt-get install/update/upgrade
    - **RaspBian** – RaspberryPi
    - **Ubuntu** – fork Debianu s různými x-managery
      - **XUbuntu** – LXDE
      - **KUbutntu** – KDE
      - **Ubuntu** – Gnome nebo Unity
    - **XBMC** – především na multimediální "krabičky"
  - **ArchLinux**
    - Např. pro BeagleBone, RPi
  - **OpenWRT** – především na síťových routerech
  - **Yocto**
    - Určené pro konkrétní HW, typicky nemá nastavené repozitory pro balíčkovací systém
    - Předpokládá se tvorby "obrazu" pro Flash paměť včetně příslušných ovladačů a provozního SW
    - Defaultně v Intel-Galileo, používaný také např. v Kontron (x86 HW)
- Balíčkovací systémy vzájemně přímo nekompatibilní, existují konverzní utility
  - Většina "hlavních" repozitory obsahuje stejný SW, drobné rozdíly ve verzích
- Z bezpečnostních důvodů by uživatel **root** neměl být přímo přístupný
  - Nutno použít příkaz sudo nebo sudo su
  - Řada distribucí to nerespektuje ☹

Embedded Linux a GPIO

- V Linuxu "je všechno soubor"
  - Také HW věci jako GPIO, SPI, PWM
  - Vzhledem k "multiuser" charakteru systému si musí konkrétní aplikace (konkrétního uživatele) daný HW prostředek "přisvojit"
  - Po použití nutno prostředek uvolnit
  - Jiný proces při přístupu hlásí chybu
  - Na HW typicky může jen root nebo uživatel s potřebnými právy (např. pomocí sudo)
- Nativní programovací jazyk je C/C++
  - Nutno překládat na cílovém HW nebo mít na PC instalovaný potřebný "toolchain" = překladač pro daný HW (např. ARM)
  - Možné použít Javu, Python, ... (po přihrání příslušných knihoven)
  - C# a .Net použitelné díky projektu Mono
    - Možno spouštět EXE (a DLL knihovny) přeložené na např. na PC s Visual Studiem
- Nastavení GPIO pinu jako výstupní = v principu se zapisuje do souborů
  - Konkrétní GPIO je nutno "exportovat" = vyhradit pro "sebe"
    - Objeví se adresář /sys/class/gpio/gpioX/ se "pseudo-soubory" direction a value
  - Směr určen zapsáním int/out do příslušného direction
  - Hodnota výstupu může být 0/1, pro vstup se přečte 0/1
  - Nezapomenout uvolnit pomocí zapsání čísla pinu do unexport
  - Samozřejmě je možné si "zabrat" více GPIO podle potřeby
- Příklad pro HW IntelGalileo s distrem Debian na microSD (cesty se podle distribuce mohou lišit)
  - GPIO 3 má připojenou LED na desce
  - Pozor na různé číslování vývodů na procesoru (= hodnoty pro GPIO) a dle popisu desky

```
root@galileo:/home/user# echo 3 > "/sys/class/gpio/export"
root@galileo:/home/user# echo out > "/sys/class/gpio/gpio3/direction"
root@galileo:/home/user# echo 1 > "/sys/class/gpio/gpio3/value"
root@galileo:/home/user# echo 0 > "/sys/class/gpio/gpio3/value"
root@galileo:/home/user# echo 3 >  "/sys/class/gpio/unexport"
root@galileo:/home/user#
```


Intel® Galileo Board Fab D
September/2013
IOREF Jumper selects 3.3V or 5V Shield Operation
ICSP
IO12 1 2 5V
IO13 3 4 ~IO11
RESET 5 6 GND
5V Brick Power Supply
Three-Output Voltage Regulator
VTT Regulator
FETs for S0/S3 states
POWER
IOREF 2
RESET 3
3.3V 4
5V 5
GND 6
GND 7
VIN 8
3.3V
VIN
5V
All GPIO PWM provided by a single I2C IO Expander
DDR3 x8
256 MB
DDR3 x8
ANALOG IN
A0 1
A1 2
A2 3
A3 4
A4 5
A5 6
MUX
MUX
GPIO PWM
COIN HDR
JTAG
I2C
ADC
SPI
FLASH
LSPI
DEDIPROG
25MHz
Clocks Out
PCIe*
Mini-PCIe*
Intel® Quark SoC X1000 Application Processor
SPI1
GPIO PWM
MUX
GPIO LED
32.768 kHz
GPIO (INT0/1)
GPIO PWM
MUX
MUX
10 SCL
9 SDA
8 AREF
7 GND
6 IO13
5 IO12
4 ~IO11
3 ~IO10
2 ~IO9
1 IO8
8 IO7
7 ~IO6
6 ~IO5
5 IO4
4 ~IO3
3 IO2
2 TX →
1 RX ←
DIGITAL (PWM~)
Host USB 1
SDIO
RMII [0]
Host USB 0
Client USB
UART 0
UART 1
PHY
RS-232 XCVR
3.3V <-> 5V Level Translation provided on board between all SoC IO and Shield Headers
Micro SD Connector
Host USB
Client USB
10/100 Ethernet
3 pin Jack (Not Audio)
Intel Galileo

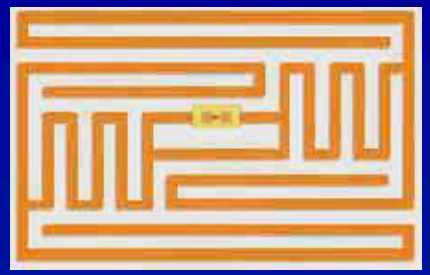
Prameny a odkazy

- http://iqjar.com/jar/an-overview-and-comparison-of-todays-single-board-micro-computers/
- http://elinux.org/Main_Page
- http://robodoupe.cz/2014/esp8266-internet-veci-prichazi/

- http://en.wikipedia.org/wiki/Comparison_of_Linux_distributions
- http://en.wikipedia.org/wiki/PackageKit

- http://www.linuxsoft.cz/article.php?id_article=1953
- http://elinux.org/RPi_Low-level_peripherals
- http://www.raspi.cz/

- http://www.zive.cz/bleskovky/intel-galileo-podrobne-detaily-o-quarku-x1000-na-desce-arduino/sc-4-a-170852/default.aspx
- http://www.intel.com/content/www/us/en/education/university/galileo-for-universities.html
- https://downloadcenter.intel.com/Detail_Desc.aspx?DwnldID=24355&lang=eng&ProdId=3777
- http://www.intel.com/content/www/us/en/embedded/products/galileo/galileo-g1-schematic.html?wapkw=galileo
- http://www.intel.com/content/www/us/en/library/viewmore.results.html?prTag=rintelproduct:inteldesktopboardsandkits/intelgalileoboardsandkits#mTag%3Drresourcetype%3Aguide%26prTag%3Drintelproduct%3Ainteldesktopboardsandkits%2Fintelgalileoboardsandkits



Technologie RFID

- Radio Frequency Identification
- Systém pro automatickou identifikaci pomocí RFID tagů
  - transponder = Transmitter-responder
  - tag je na nebo uvnitř objektu
  - přenos zajišťován radiovými vlnami
- Nosná frekvence 125kHz, 134kHz, 13.56MHz
- Další 433/868MHz, mikrovlné 2.4-5GHz, 3.1-10GHz
- 96-bitový unikátní identifikátor
  - EPC = Electronic Product Code
  - alternativně 64-bitový, možný i 128-bitový
- Read-only nebo read-write tagy
- Běžná realizace antény (RF část)
  - nízké frekvence - cívky na feritu
  - vysoké frekvence - plošné

Typy RFID tagů

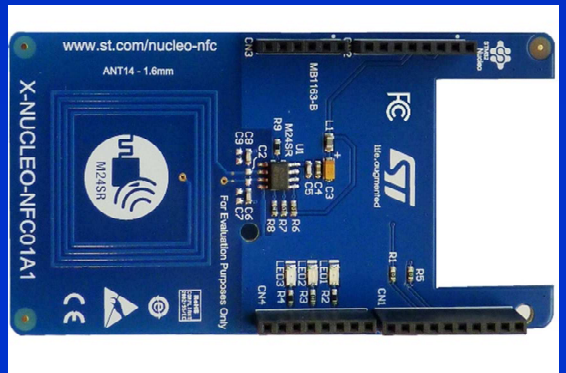
- Dělení podle způsobu napájení
  - Pasivní
    - žádné vlastní napájení
    - přijatý radiový signál z antény nabije kapacitu, která pak vystačí na odvysílání potřebné informace
    - čtecí vzdálenost od cca 10 cm (ISO 14443), max. řádově metry
  - Aktivní
    - vlastní napájení
    - až stovky metrů dosah
    - možnost vlastních funkcí - sledování teploty apod.
  - Semi-pasivní (battery-assisted)
    - baterie napájí pouze čip, ne vysílací část
    - vyšší citlivost proti pasivním, delší výdrž baterie oproti aktivním
- Dělení podle zapisovatelnosti
  - pouze čtení (class 0), programováno ve výrobě, čtení 1000tag/s
  - zapisovatelné 1+x (class 1), prog. při nasazení, čtení 200t/s
  - zápis vícenásobný (class 0+), prog. kdykoliv, 256bit, čtení 1000t/s
  - zápis vícenásobný (Gen2), prog. kdykoliv, 256bit, 1600t/s

Využití RFID

- E-passport = elektronický pas
  - poprvé Malajsie 1998 (ukládá "razítka")
  - 2005-6 US - obsahuje i digitální foto
- Platby mýtného apod.
  - aktivní RFID tagy
- Sledování zboží
  - označení knih v knihovnách
  - pohyb zboží ve skladu
- Automobily
  - identifikace startovacího klíčku
- Označování zvířat
  - tag ve formě skleněné tubičky implantován
    - typicky pod kůži
  - evidence domácích zvířat (např. psů)
  - užitková zvířata

Další aplikace RFID

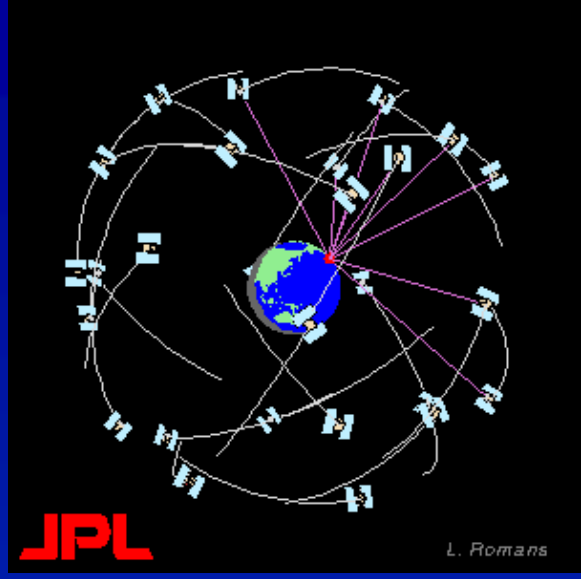
- Miniaturizace
  - 2009 (Bristol Univ.) – micro-transponder přilepen na mravence – výzkum chování
- Visačky pro sledování zavazadel na letišti
- Síť distribuovaných senzorů – výzkum bouřek/větrných smrští
- Sportovní akce – identifikace závodníků, měření časů u hromadných akcí
- NFC (Near Field Communication) vychází z RFID



GPS

- Gde Právě Sem ?
- Global Positioning System
  - provozován US ministerstvem obrany
  - původní jméno Navstar
  - přesnost metry, možná až centimetry při použití DGPS (diferenciálního = nutný referenční přijímač se známou polohou)
  - 24 družic (včetně 3 záložních)
- Určení polohy
  - zjištění vzdálenosti od známé polohy družic pomocí zpoždění radiových vln
  - družice má extrémně přesné hodiny
  - vysílají i svoji polohu (parametry dráhy)

GPS - II



- Družice
  - obíhají po 6 drahách se sklonem 55º posunutých o 60º
  - výška 20 200km
  - vysílací kmitočty (L1-L5, 1000-2000MHz)
    - C/A - civilní kód
    - P/Y - šifrovaný vojenský kód
    - výsledky sledování atomových pokusů a měření ionosféry
- Bitová data
  - 1 bit trvá 20ms
  - další jednotky slovo, podrámec a rámec (30 sec)
  - slova zabezpečena Hammingovým kódem
- Ústředí řízení GPS - Navstar Headquarters
  - letecká základna Los Angeles, Kalifornie

GPS - III

- Sirf Technology - jedna z prvních firem vyrábějící GPS moduly
  - SirfStarIII – první dostupný chipset
- SirfStarV – základ platformy SirfFusion (+ WiFi, MEMS, radio) + podpora dalších navigačních systémů
  - GLONASS (Глобальная навигационная спутниковая система)
    - plně v provozu od 2010 (Rusko), resp. 2011 (světově)
  - Galileo – evropská varianta
    - přesnost pod 1m
    - všechny satelity plánovány do 2019
    - administrativa – Praha
    - pozemní řídící střediska – Mnichov, Fucino (IT)
  - Compass – BeiDou Navigation Satellite System (北斗衛星導航系統)
    - čínská varianta GPS
    - 2012 oblast Asie-Pacifik, do 2020 celosvětově

GPS - III

- Získané informace podle počtu přijímaných družic
  - alespoň jedna - přesný čas
  - alespoň 3 - poloha
  - alespoň 4 - poloha a výška
- Užití výstupu GPS přijímače
  - nejčastěji ve formátu NMEA 0183
  - sériová data, textová podoba, organizováno do "vět" uvozených $, na konci kontrolní součet
  - typicky 4800b/s, možnost nastavení parametrů

```
$GPGSA,A,3,29,26,22,09,07,05,04,,,,,,1.7,1.0,1.4*30
$GPGSV,3,1,11,09,84,297,41,05,48,256,45,07,38,059,41,26,22,178,41*74
$GPGSV,3,2,11,24,13,063,00,14,12,324,00,30,12,251,00,22,12,286,38*78
$GPGSV,3,3,11,29,10,173,35,04,09,105,30,18,06,254,00*46
$GPRMC,170138.615,A,4912.2525,N,01635.0378,E,0.04,16.43,280705,,*32
$GPGGA,170139.615,4912.2526,N,01635.0378,E,1,07,1.0,357.5,M,43.5,M,0.0,0000*7D
```

Související odkazy

- http://cs.wikipedia.org/wiki/RFID
- http://en.wikipedia.org/wiki/Radio-frequency_identification
- http://automatizace.hw.cz/rfid-senzory-soucasna-situace
- http://www.odbornecasopisy.cz/automa/2007/au070701.htm
- http://nearfield.cz/co-je-nfc
- http://cs.wikipedia.org/wiki/GPS
- http://www.kh-gps.de/nmea-faq.htm
- http://en.wikipedia.org/wiki/SiRF
- http://www.abclinuxu.cz/serialy/gps-a-komunikacni-protokol-nmea
- http://www.abclinuxu.cz/clanky/ruzne/gps-a-komunikacni-protokol-nmea-3-dekodovani-dat
- http://www.esa.int/Our_Activities/Navigation/The_future_-_Galileo/What_is_Galileo
- http://www.czechspaceportal.cz/2-sekce/agentura-gsa/sidlo-gsa-v-praze/
- http://en.wikipedia.org/wiki/Beidou_navigation_system
- http://cs.wikipedia.org/wiki/Global_System_for_Mobile_Communications
- http://www.rfid-epc.cz/download/prezen/RFIDWorkingGroup-UvodDoTechnologie.pdf

8. přednáška
Připojování periférií k PC + různé 2

- Invoke a PInvoke

- Samostatná zařízení
  - Sériové a paralelní porty
  - USB
  - Ethernet
  - Firewire
- Vestavěná zařízení
  - ISA sběrnice
  - PCI sběrnice
  - PCIe
  - PCMCIA (PC-Card), CardBus, ExpressCard
- Praktické zkušenosti (KAE)
  - sériový, paralelní port, USB2serial, USB host
  - ISA, PCI ve VHDL

Invoke

- .NET omezuje přístup k vizuálním prvkům
  - může jen thread, který prvek vytvořil
  - typicky je to hlavní thread aplikace
  - ostatní thready vytváří např. každá asynchronní akce
- Při práci s prvkem nutno napřed otestovat vlastnost `InvokeRequired`
  - nastavena **true** pro „cross-thread" volání
- Funkce bez parametrů, příp. s globálními parametry

```
this.BeginInvoge(new MethodInvoker(this.funkce));
```

- Vytvořit delegáta a zavolat s parametry

```
delegate void AddProtokolDelegate(String s, int x);
void AddProtokol(String s, int x)
{
  if (this.InvokeRequired)
  {
    this.BeginInvoke(new AddProtokolDelegate(AddProtokol),
      new object[] { s, x });
    return;
  } // if (this.InvokeRequired)
  ...
}
```

- Možnost použití systémových typových delegátů `Action<...>`
  - od .NET 4 výše

PInvoke

- Součástí .NET FW je řada objektů (tříd) reprezentujících funkce systému Win32
- Ostatní funkční volání je možno provést přímým zavoláním funkce v DLL knihovně
- Funkce musí být „dekorována“ pomocí DllImport a korektně nastaveny parametry
- Podrobnosti o konkrétní funkci viz. MSDN help pro Win32
- Parametry vycházejí z C/C++ syntaxe
- Příklady a formální předpis např. http://pinvoke.net

```
[DllImport("user32.dll", EntryPoint = "GetDesktopWindow")]
public static extern int GetDesktopWindow();
[DllImport("User32.dll")]
public static extern Boolean EnumChildWindows(int hWndParent, Delegate lpEnumFunc, int lParam);
```

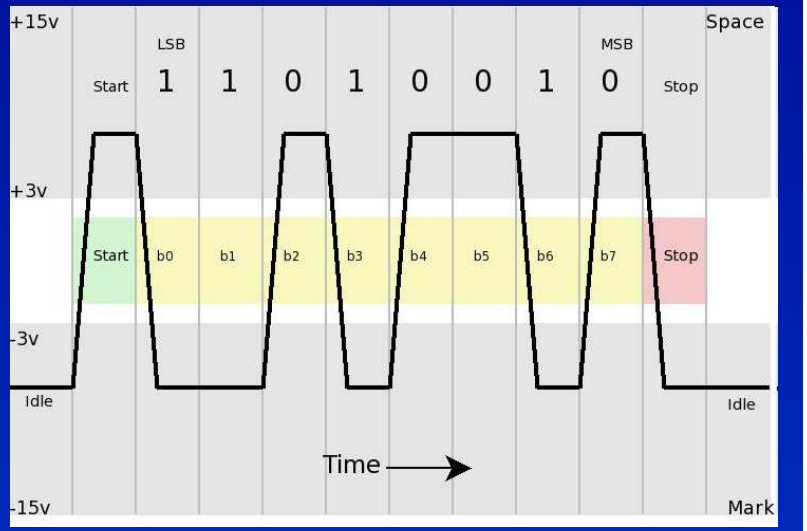
Odkazy

- http://www.manualy.net/article.php?articleID=13
- http://pinvoke.net
- http://www.codeproject.com/KB/cs/AOPInvokeRequired.aspx

Sériové porty

- Nejjednodušší možnost připojení k PC
- Pro moderní PC bohužel "obsolete" (=zastaralý), notebooky již ani nemají konektory
- Prakticky každý jednočipový mikropočítač má:
  - UART (= "Universal Asynchronous Receiver/Transmitter")
  - kompatibilní s protokolem RS-232
  - někdy též USART (tedy + "Synchronous" režimy)
- Stačí 2 datové vodiče (tam a zpět) + GND

Sériový přenos



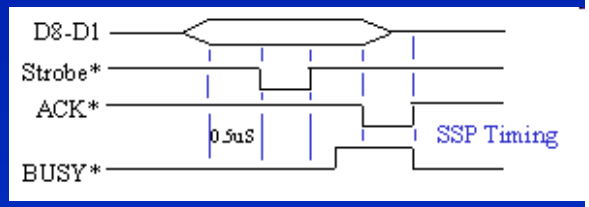
- Fyzická reprezentace
  - napěťové úrovně
  - log. 1 = -3V až -15V
  - log. 0 = +3V až +15V
- Rámec dat
  - uvozen START-Bitem = log. 0
  - ukončen STOP-Bitem o délce 1, 1.5 nebo 2
  - datových bitů 5-8 + případná parita
- Typické přenosové rychlosti
  - dané dělením základního kmitočtu 4.77MHz (např. první PC)
  - 110, 300, 1200, 2400, 4800, 9600, 19200, 28800, 38400, 57600, 76800, 115200 (230400 - 2764800) bit/s
  - nezbytná přesnost 5% = nesmí se lišit o víc než ½ bitu na 10 bitů (8 dat + 2 řídící)

Sériový přenos - záludnosti

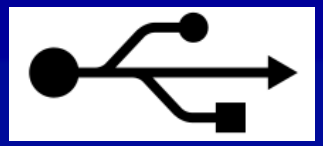
- Typicky je v PC zařazen FIFO buffer
  - při vysílání je přerušení při předání bytu do výstupní fronty = problém s časováním konce posledního znaku
  - při příjmu je často nutno vyčíst více znaků najednou
- Nejčastější praxe je vypínat řízení toku (HandShake)
- Některé sériové řadiče v PC vyžadují fyzické spojení signálů RTS-CTS a DSR-DTR, jinak nekomunikují
- Konvertory USB-serial principielně nedokáží zajistit stejné časování
  - problémy s programátory a dalším spec. SW a HW pro práci s jednočipy
  - zatím nejkompatibilněji se chovají FTDI čipy
  - pro "vzpurné" aplikace pomáhá PCMCIA karta s porty

Paralelní port

- Jednoduché připojení periférií paralelně
- V moderních PC už dlouho "obsolete"
- Základem sběrnice 8-bitů dat a řídící signály
  - /Strobe (out) = platnost dat
  - /Busy (in) = zařízení pracuje, neposílat data
  - dodatečné - Ack, PaperOut, Select, AutoFeed



- Základní přenos cca 360kB/sec
- Rozšíření režimy s obousměrnou datovou sběrnicí
  - viz IEEE1284 specifikace
  - SPP - half-duplex na základních 8 datových vodičích
  - EPP/ECP - další zvyšování rychlosti - až 2.5MB/sec
  - komplikovaný HandShake podpořený v HW základní desky



USB - přehled

- Určeno jako náhrada všech periferních sběrnic PC
- Vlastnosti
  - zařízení možno připojit za chodu
  - max. 127 zařízení na sběrnici
  - max. délka kabelu 5m
  - obsahuje i 5V napájení (100mA, po ohlášení až 500mA)
- Varianty
  - USB 1.1 – přenosy 1.5 nebo 12Mbit/s
  - USB 2.0 – přenosy až 480Mb/s u Hi-Speed zařízení
  - USB 3.0 – až 4.8GB/s, 900mA, od 2010
- Každé zařízení má identifikaci (2 x 16-bit hodnota)
  - VID (Vendor ID) = kód výrobce, přidělován USB Consorciem
  - PID (Product ID) = číslo produktu
- Host Controller (Device = HCD) řídí přenosy po sběrnici

USB - přehled - II

- Data se přenášejí v paketech 8 až 256 bytů
- Přenos v rámcích délky přesně 1ms
- Typy přenosů na USB
  - řídící přenos (Control)
    - konfigurace a řízení zařízení
    - vysoká priorita
  - přenos při přerušení (Interrupt)
    - pro pravidelný přenos malého množství dat (myš, klávesnice)
    - přerušení řešeno SW, master se pravidelně dotazuje (10ms)
  - hromadný přenos (Bulk)
    - pro velké množství dat, kde není čas kritický (scanner, tiskárna)
    - nízká priorita na sběrnici
    - provádí se korekce chyb
  - izochronní přenos (Isochronous)
    - přenos dat definovanou rychlostí, nesmí dojít k výpadkům
    - neprovádí se korekce chyb (zvukové karty apod.)

USB - virtuální COM port

- Nejjednodušší využití USB pro připojení externích zařízení
- Na straně periférie běžný sériový port
- Omezení
  - nelze zaručit přesné časování
  - v PC musí být nainstalován ovladač zařízení "USB-to-Serial" konkrétního výrobce čipu
  - některé starší SW odmítají komunikovat na portu COM > 4 (či dokonce 2)
  - pro COM >= 10 nutno ve Windows speciální „název“ – „\\\\.\\COMxx”

USB - vlastní Slave

- Jednočipový mikropočítač přímo s řadičem schopným provozu jako USB Slave
  - optimální využití možností procesoru
  - možné problémy perspektivností obvodu a tím i dlouhodobější dostupností pro výrobu/servis
- Využití specializovaného obvodu
- SW řešení
  - např. pro procesory AVR

    http://www.obdev.at/products/avrusb/index.html
- Problémy s ovladači na hostitelském PC
  - vhodné řešení je implementovat standard "Mass Storage" - obdoba Flash disků

USB - vlastní Host

- Na straně PC existuje projekt libusb
  - primárně pro Linux
  - existuje i 32-bit Windows port
- Pro jednočipové mikropočítače typicky využití "hotových" čipů
  - problém s dostupností v kusovém množství
- Současné výkonné (nejčastěji 32-bitové) mikroprocesory již v některých typech obsahují USB-Host
  - např. PDA s PXA270 či lepším
  - některé ARM7 nebo ARM9 a lepší nebo Cortex-Mx

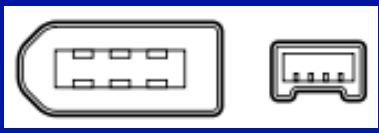
Ethernet

- Rozšiřuje se počet zařízení s Ethernet portem (známý konektor RJ-45 - "kostka")
- Výhodou je ověřené řešení pro rozsáhlé množství platforem
- Každé zařízení má jedinečnou identifikaci - MAC adresu (48-bitová)
- Přenosová rychlost 10/100Mbit nebo 1GBit
- Základní řešení složeno ze 2 částí
  - PHY
    - odpovídá fyzické vrstvě ISO/OSI
    - zajišťuje přenos paketů Ethernetu
  - MII - "Media Independent Interface"
    - zpracovává pakety PHY
    - obsahuje MAC adresu
- Hotové řadiče ve formě obvodu
  - Připojeny k procesoru pomocí UART, SPI
  - Např. produkty WizNet – W5100 umí 4 TCP/IP spojení (viz. cv.)

Ethernet pro aplikace

- Zařízení s Ethernetem získává IP adresu
  - buď pevně nastavena při konfiguraci
  - nebo implementuje DHCP protokol pro získání IP adresy z nadřízeného zařízení (např. NAT na routeru)
- Možno přistupovat z libovolného počítače/zařízení s IP adresou
- Je vhodné implementovat standardní protokoly
  - UDP - jednoduchý, není zaručeno doručení
  - TCP/IP - náročnější SW (nutný TCP/IP stack)
  - FTP - vhodné pro přenos uložených dat ve formě souborů
  - HTTP
    - optimální varianta pro využití prohlížeče u klienta
    - vyšší režie implementace - nutno zpracovat hlavičky apod.
    - vhodné realizovat nějakou "dynamičnost" stránek odpovídající charakteru sledovaného procesu
- Existují řešení i pro 8-bitové procesory
  - problémy s výkonem a dostupnou pamětí (prakticky nerealizovatelné na x51 v jazyce C)

FireWire



- Též i.Link (Sony), resp. IEEE1394
- Typicky pro digitální kamery nebo externí datové nosiče
- Základní vlastnosti
  - Periférie možno připojovat za chodu
  - 63 periférií na sběrnici
  - možnost komunikace mezi zařízeními
  - malá zátěž CPU v PC
- Přenosové rychlosti a vzdálenosti
  - 100, 200, 400Mbit/sec, max. 4.5m kabel (FW400)
  - 800M - 3.2Gbit/sec, kabel až 100m (FW800 = 1394b)

Sběrnice ISA

- Základní sběrnice v PC
  - PC+PC/XT - 8bitová, PC-AT - 16-bitová
  - typická rychlost 8MHz
  - vychází z architektury procesoru 8086
- Periférie mapovány do paměťového prostoru procesoru (porty i paměť)
- Možnost využít DMA kanálu
- Funkčnost Plug'n'Play (PnP)
  - Doplněna později = vyžaduje podporu HW, BIOSu a OS
- Jednoduchá realizace
  - stačí 1 GAL 16V8 pro dekódování adres a signálů
  - 74x245 jako oddělovač sběrnice pro vstup
  - 74x574 jako registry pro výstup
- V současných PC neexistuje ...

Sběrnice PCI

- Standard nejen v PC
  - přenosová rychlost 133MB/s
  - 32-bitová sběrnice, synchronní přenos
  - aktuální verze 2.1
  - možné varianty 5V a 3.3V (různé umístění zámku na slotu)
  - alternativně 64-bitové rozhraní nebo 2x/4x takt
    - především u serverů
- PnP - zařízení sdělí správci periférií požadavky na paměť a porty, dostane přiděleny fyzické adresy
- Praktické řešení
  - založené na VHDL blocích pro FPGA řešení - cena!
  - specializované čipy - typicky určené pro převod na jednodušší rozhraní
  - vlastní vývoj - náročné, nutno splnit standardy OS (např. Microsoft WHQL)

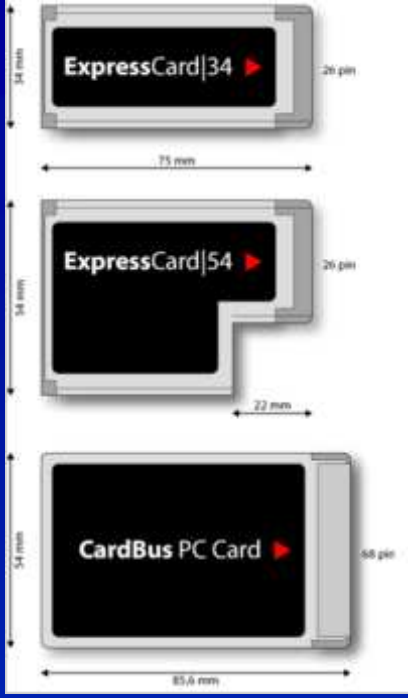
Sběrnice PCI Express

- Implementace PCI pomocí sériové komunikace
  - založena na vyhrazeném 1-bitovém spojení nazvané "lane"
  - rychlost lane je 2.525Gbit/s (resp. 2x pro verzi 2)
  - fyzicky LVDS (Low Voltage Differential Signaling) zvlášť pro přenos tam a zpět (celkem 4 vodiče)
- Spojení 2 PCIe zařízení se nazývá "link"
  - možno spojovat na 2, 4, 8, 16 nebo 32 lane
  - spojení řeší „switch“ mezi libovolnými 2 zařízeními
  - aplikačně zajištěna kompatibilita s tím, že se nevyužije potenciál "výkonnějšího" zařízení
- Vlastní vývoj na KAE zatím nerealizován (?)

Sběrnice PCMCIA (PC-Card), CardBus

- Peripheral Component MicroChannel Interconnect Architecture
- Typické rozhraní pro starší notebooky
- Napájení 5V nebo 3.3V
  - mechanicky chráněna 3V karta proti použití v 5V slotu
- Základní velikost 85.6x64mm s různou tloušťkou
  - Type I - 3.3mm, 16-bitové rozhraní
  - Type II - 5.0mm, 16/32 bitová
  - Type III - 10.5mm, 4 řady kontaktů
- Aktuální varianta - CardBus (= PCMCIA 2.1), plně 32-bitová, sloty často kompatibilní se starší PCMCIA

Sběrnice ExpressCard



- Náhrada CardBus
- V novějších NTB
  - počínaje Intel Core2Duo
  - vyjímečně kombinace s CardBus (profi ntb)
- Vychází z PCIe
  - využívá vlastního řadiče pro snížení zátěže procesoru PC
  - max. rychlost 2.5Gbit/s proti 1066Mbit/s CardBus
- Nekompatibilní s CardBus !!

Zajímavé odkazy

- http://en.wikipedia.org/wiki/UART
- http://en.wikipedia.org/wiki/RS-232
- http://www.freebsd.org/doc/en_US.ISO8859-1/articles/serial-uart/
- http://en.wikipedia.org/wiki/RS-232
- http://en.wikipedia.org/wiki/D-subminiature
- http://en.wikipedia.org/wiki/IEEE_1284
- http://pinouts.ru/ParallelPortsCables/Printer_pinout.shtml
- http://computer.howstuffworks.com/parallel-port.htm
- http://cs.wikipedia.org/wiki/Usb
- http://en.wikipedia.org/wiki/Usb
- http://www.usb.org/home
- http://libusb-win32.sourceforge.net/#documentation
- http://www.lvr.com/usbcode.htm
- http://www.ftdichip.com/FTProducts.htm
- Přednášky předmětu KAE/DZD, resp. KAE/RIS
- http://en.wikipedia.org/wiki/Ethernet
- http://en.wikipedia.org/wiki/PHYceiver
- http://en.wikipedia.org/wiki/Media_Independent_Interface
- http://cs.wikipedia.org/wiki/Firewire
- http://en.wikipedia.org/wiki/Firewire
- http://en.wikipedia.org/wiki/Industry_Standard_Architecture
- http://en.wikipedia.org/wiki/Peripheral_Component_Interconnect
- http://pinouts.ru/Slots/PCI_pinout.shtml
- http://www.root.cz/clanky/podrobnejsi-popis-sbernice-pci/
- http://www.tedia.cz/cz/index.htm
- http://cs.wikipedia.org/wiki/PCI-Express
- http://en.wikipedia.org/wiki/PCI_Express
- http://www.root.cz/clanky/interni-sbernice-pci-express/
- http://www.svethardware.cz/technologie-sbernice-pci-express/11606
- http://en.wikipedia.org/wiki/Cardbus
- http://www.pcmcia.org/about.htm
- http://en.wikipedia.org/wiki/ExpressCard

9. Přednáška
IoT, komunikace, P4.0 + drobnosti

- Životní cyklus aplikace
  - WinForms
  - WPF
  - Android
  - (Windows Phone)

- MicroPython
  - Další možnost využití vysoko-úrovňového jazyka pro mikrokontroléry

---

- Espruino
  - Javascript pro mikrokontroléry

- IoT – Internet of Things
- Komunikace pro delší vzdálenosti
- Průmysl 4.0

Životní cyklus aplikace

- Moderní API jsou založená na textovém (XML) popisu vzhledu a kódem realizujícím třídy objektů
- Životní cyklus mobilních aplikací podřízen minimalizaci spotřeby a zdrojů
  - omezený multitasking
  - app v paměti „spí“, může být odstraněna v případě nedostatku paměti
  - speciální mechanismy při komunikaci „na pozadí“

WinForms

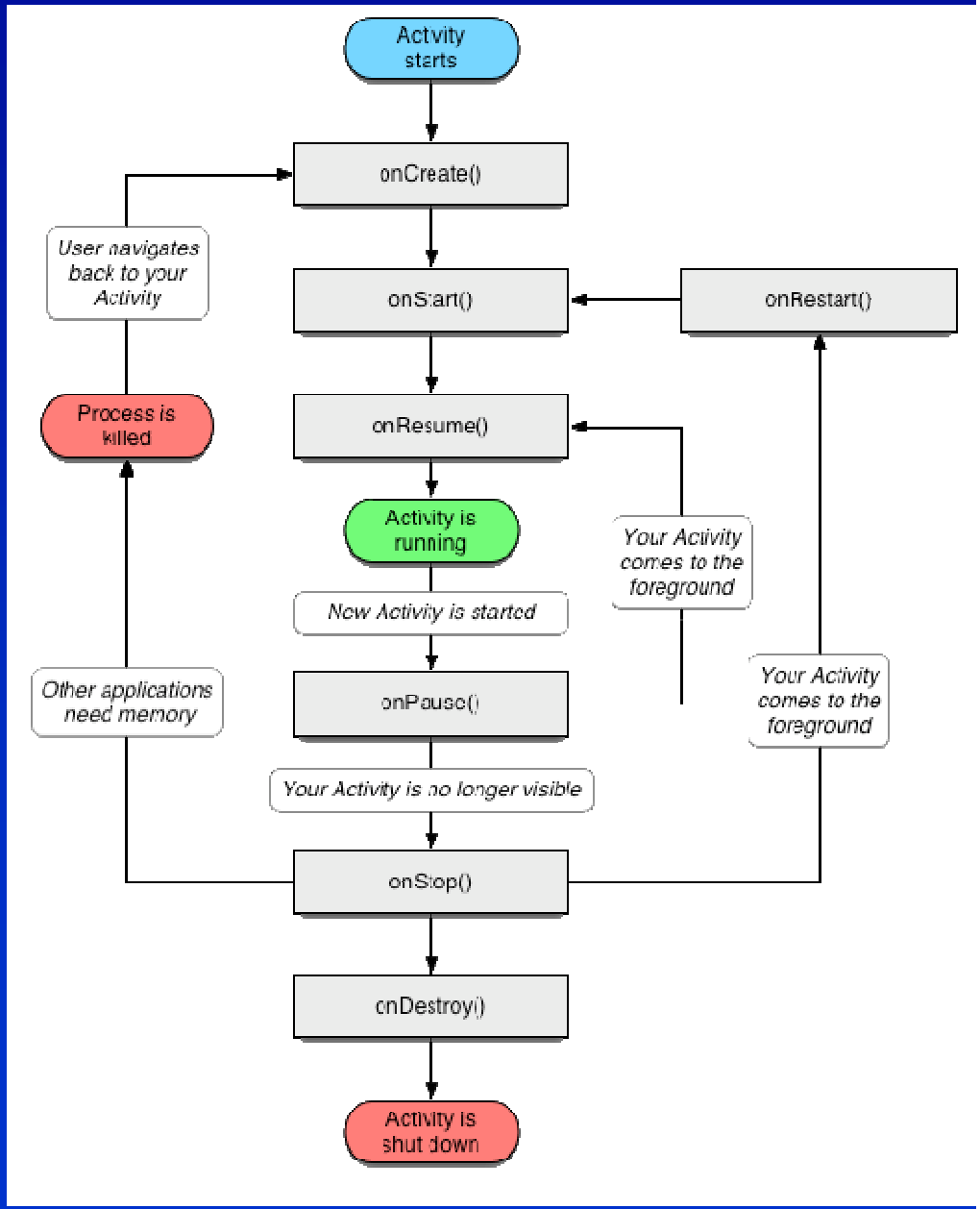
- Spouští se WinMain (C/C++)
- Jedna třída musí mít právě jednu statickou funkci Main (C#)
- Vizuální popis je realizován v kódu, designer je pouze rozhraní mezi IDE a kódem
  - v C# je soubor xxx.designer.cs
  - obsahuje funkci InitializeComponents()
- Události zpracovává jedna funkce a volá další části
  - app může běžet „na pozadí" = pravý multitasking

WPF – Windows Presentation Foundation

- Popis vizuálních prvků vychází z XML
  - jazyk XAML
  - snadná implementace animací, efektů
    - Časování pomocí objektu Storyboard
  - orientované na Binding (=provázání vlastností a hodnot) prvků
    - vizuálních i nevizuálních
    - teoreticky méně kódování
    - složitější učení
- Celé rozhraní (UI) je vektorově orientované
- K dispozici od .Net 3

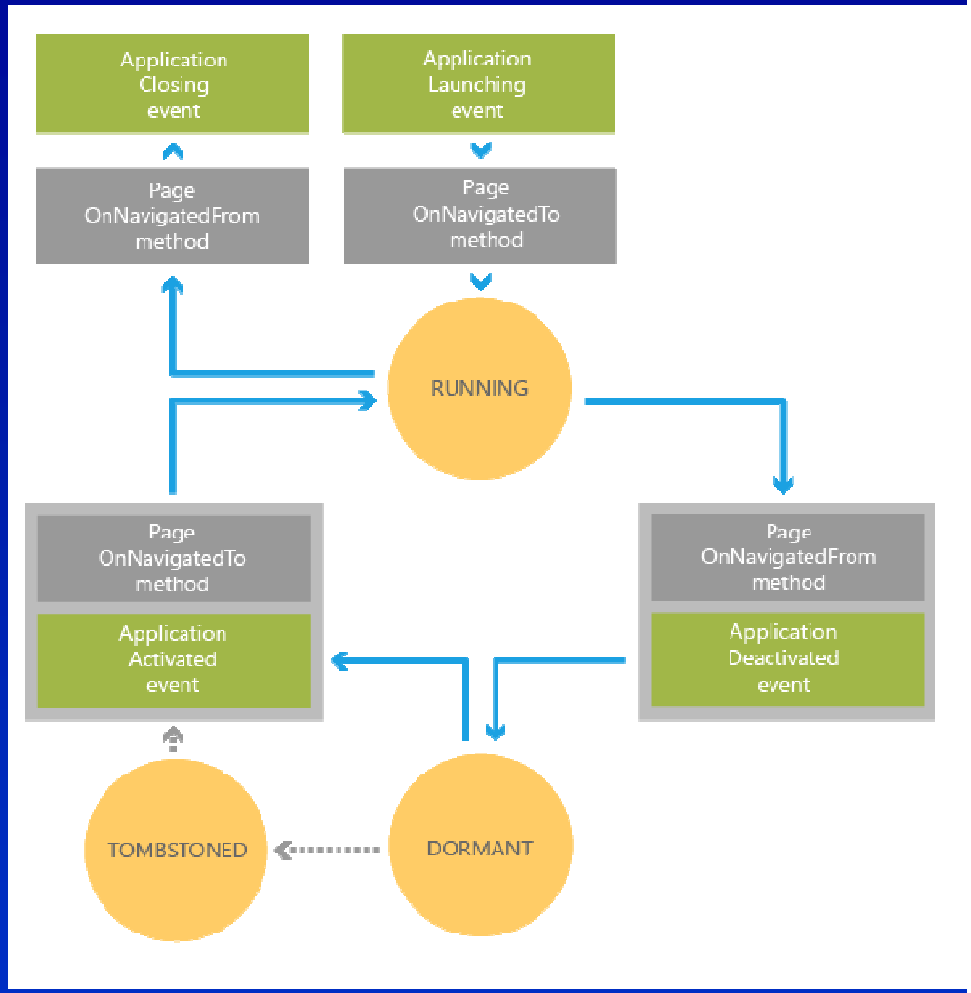
Android

- App se skládá z min. 1 Activity
  - nutno přetížit min. fukci onCreate()
  - v onResume/onPause vhodné načíst/uložit nastavení
- Možno vytvářet také
  - Dialogy
  - Fragmenty (reprezentují View v Activity)
- Popis vizuálních prvků v main.xml
  - může se lišit pro různé rozlišení/orientaci
  - typické kontejnery …Layout
    - LinearLayout
    - TableLayout
    - …



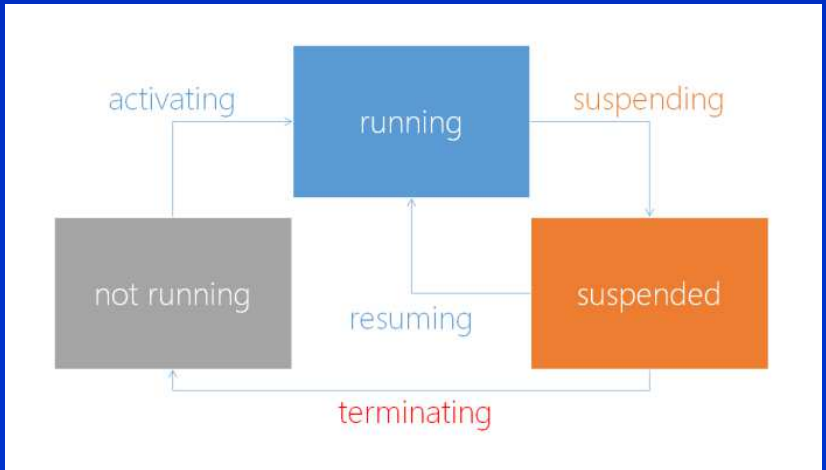
Windows Phone

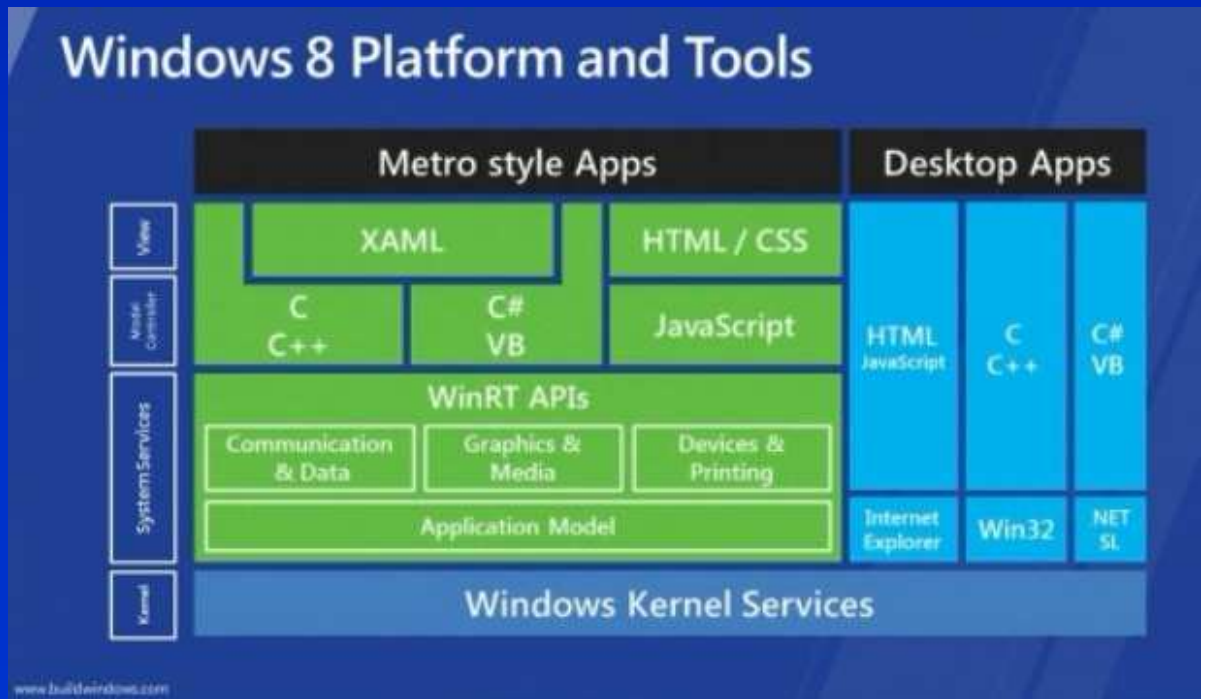
- Různé „triky“ k nahrazení multitask
  - Dlaždice
  - BackgroundAgent
  - PeriodTask
  - PushNotifikace
- Jiná filozofie UI
  - MetroDesign – doporučení vizuálu
  - definice UI v XAML
  - konfigurace v XML (Manifest)



Windows Store – UWP

- Aplikace pouze pro Win8+
- Vizuálně některé prvky/principy podobné WinPhone
  - dlaždice, …
- Programování
  - HTML+CSS+JS
  - XAML+.Net (C#, …)
  - XAML+C++

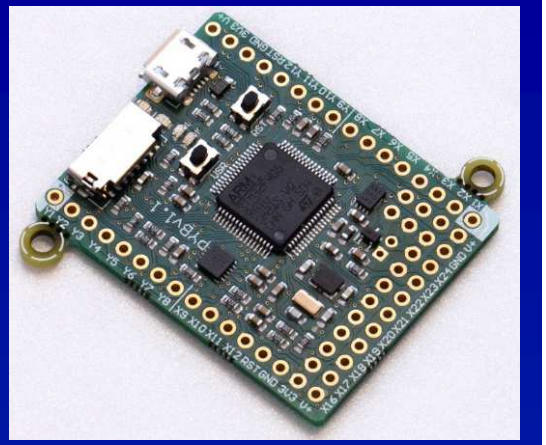




Odkazy

- https://msdn.microsoft.com/library/aa970268(v=vs.100).aspx#Controls
- http://www.zdrojak.cz/clanky/vyvijime-pro-android-dialogy-a-activity/
- http://www.winphone.cz/Intro/ZivotniCyklusAplikace
- http://www.devbook.cz/c-sharp-windows-store-aplikace

MicroPython



- **Python** je jazyk se syntaxí "ne-Céčkovou" !!

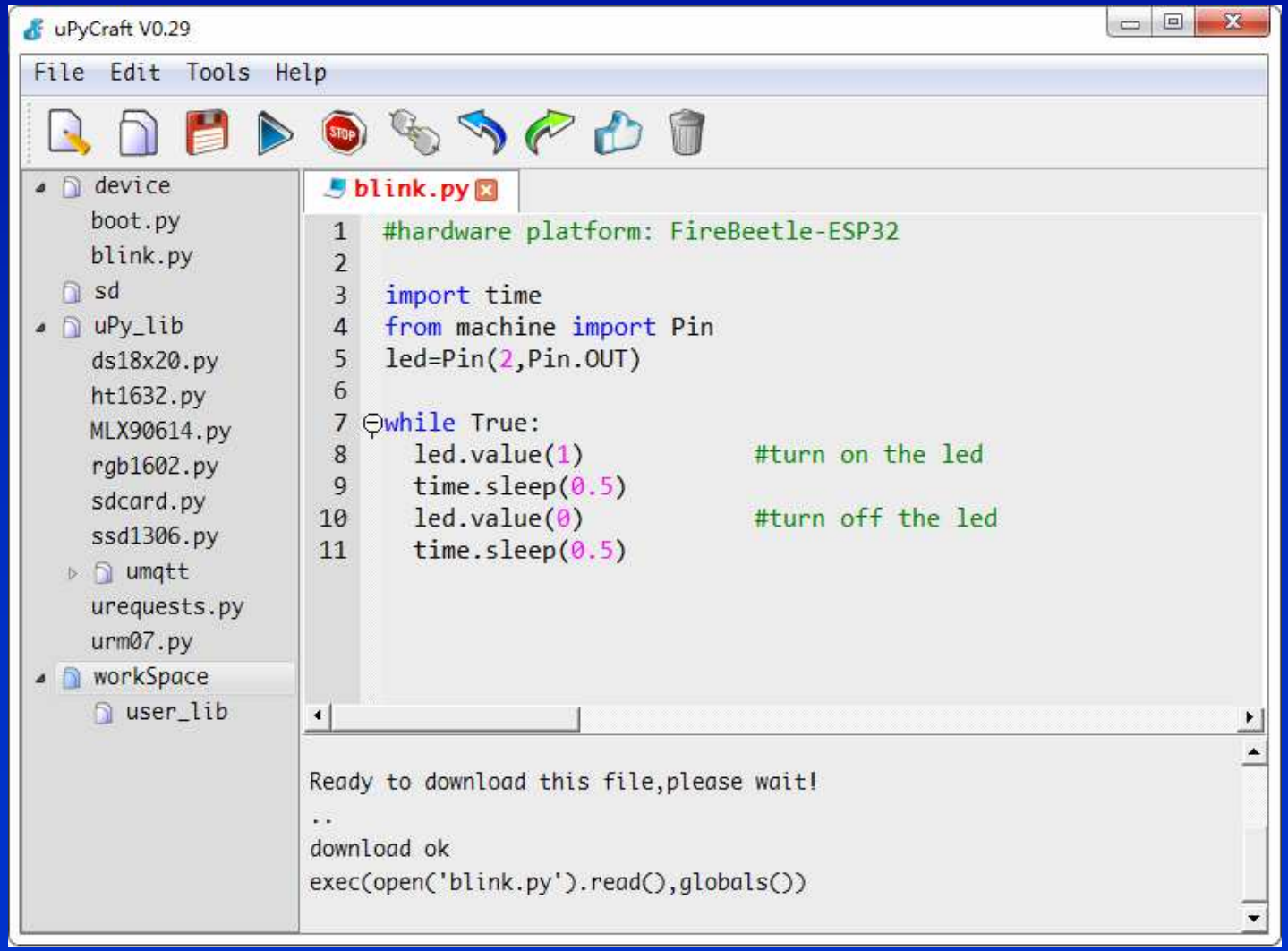
- Vzorový HW **pyboard** založený na STM32F405RG
- Portováno pro řadu F4x, ESP32
  - Dev. moduly založené na (zřejmě) ESP32 + komunikační moduly – WiFi, SigFox, LoRa, ...
- Podporuje podmnožinu objektů z "dospělého" CPython (v3)
  - Možno používat podobné "importy"
  - Doplněné HW (low-level) specifické moduly (= knihovny objektů)
- Součástí je připojení pomocí terminálu pomocí **REPL prompt** (Read Evaluate Print Loop)
  - Umožňuje výpis, vytváření a změny objektů
  - Data v RAM = možno s nimi pracovat ve skriptu (funkci) nebo manuálními příkazy (jazyka Python)

MicroPython – ukázky kódu

```
from machine import Pin, PWM, Timer

led1 = Pin(25, Pin.OUT)
led2 = Pin(26, Pin.OUT)
#led0 = Pin(2, Pin.OUT)

duty0 = 128
dir0 = 0
step0 = 32
pwm0 = PWM(Pin(2), freq=1000, duty=duty0)

def func1(t):
  led1.value(1 ^ led1.value())
  # print(led0.value())

tim1 = Timer(1)
tim1.init(period=200, mode=Timer.PERIODIC, callback=func1)

def func0(t):
  global duty0, dir0, step0

  if (dir0 == 0):
    duty0 += step0
    if (duty0 >= (1024 - step0)):
      dir0 = 1
  else:
    duty0 -= step0
    if (duty0 < step0):
      dir0 = 0

  pwm0.duty(duty0)
#  print(duty0)

tim0 = Timer(0)
tim0.init(period=50, mode=Timer.PERIODIC, callback=func0)

try:
   while True:
     pass
except:
   tim1.deinit()

   tim0.deinit()
   pwm0.deinit()
```

```
import utime

from machine import Pin,I2C
import ssd1306

i2c = I2C(scl=Pin(22), sda=Pin(21), freq=100000)
lcd = ssd1306(128,64,i2c)

lcd.fill(0)   # clear = fill black
lcd.text("Demo 2018",0,0)
lcd.show()

utime.sleep_ms(2000)

lcd.fill(0)
lcd.line(16,40,26,50,1)
lcd.line(26,50,128-26,50,1)
lcd.line(128-16,40,128-26,50,1)

lcd.rect(26,8,20,20,1)
lcd.rect(128-26-20,8,20,20,1)
lcd.fill_rect(26,18,10,10,1)
lcd.fill_rect(128-26-20,18,10,10,1)

lcd.show()

#print("CTRL+C to stop CLOCK")
while(0):
  t = utime.localtime()
  s = '{:02d}:{:02d}:{:02d}'.format(t[3], t[4], t[5])
  print(s)
  lcd.fill_rect(0,16,128,24,0)
  lcd.text(s, 0,16)
  utime.sleep_ms(200)
  lcd.show()
```

IDE – uPyCraft



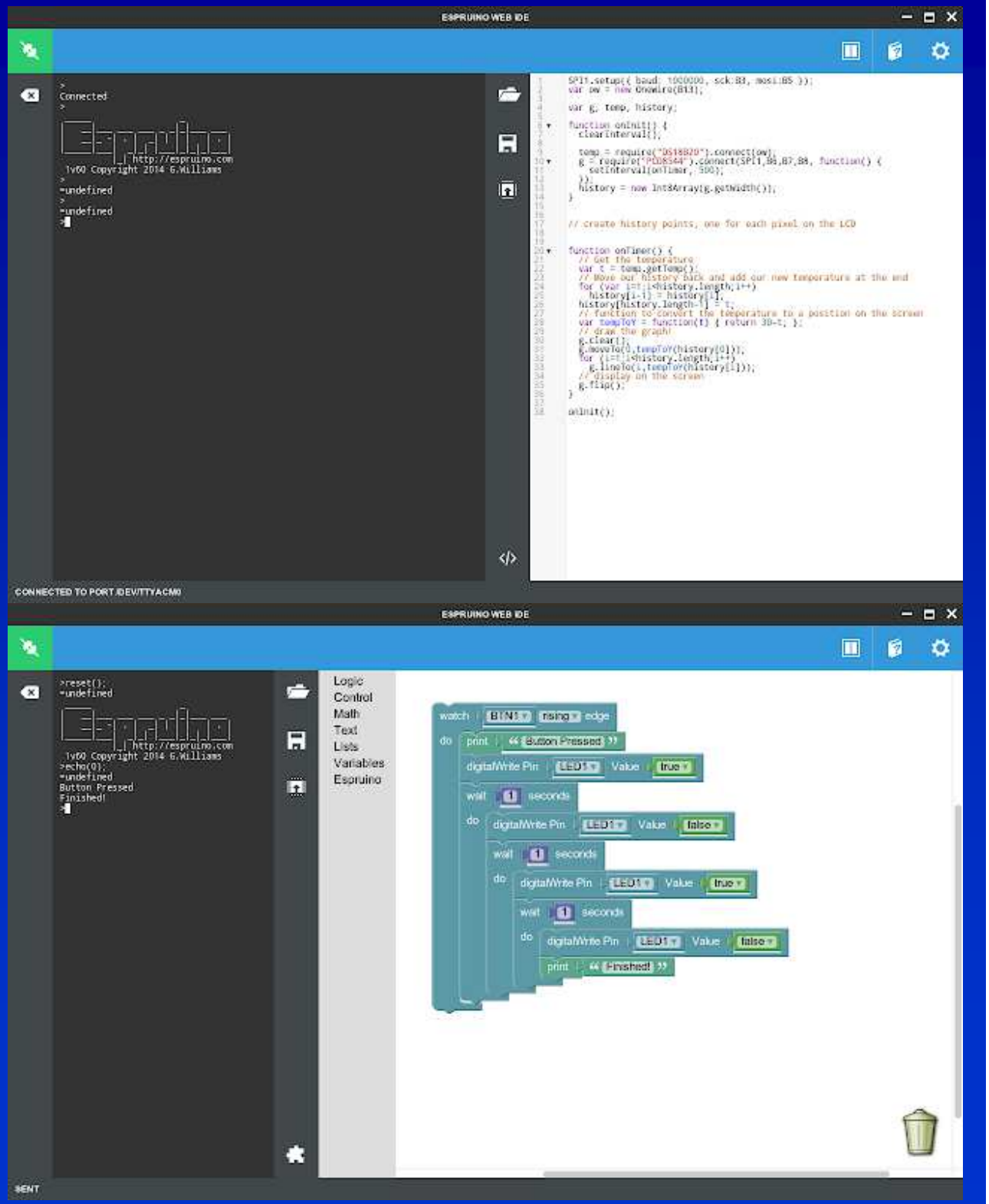
Espruino

- Programování v JavaScriptu
  - Projekt OSS
  - IDE v prohlížeči
  - Možnost debug přes GDB
  - Řada příkladů pro různé HW moduly
- Port pro Cortex-M – výchozí STM32Fx
  - Vzorová HW implementace STM32F401 a STM32F103
  - Mnoho dalších HW platforem (včetně Nucleo F401 a F411, ESP32)

```
var on1,on2;
function toggle1() {
  on1 = !on1;
  digitalWrite(LED1, on1);
}
function toggle2() {
  on2 = !on2;
  digitalWrite(LED2, on2);
}

setInterval(toggle1, 400);
setInterval(toggle2, 456);
```

```
Pin.prototype.startFlashing = function(period) {
  if (Pin.intervals==undefined) Pin.intervals = [];
  if (Pin.intervals[this])
    clearInterval(Pin.intervals[this]);
  var on = false;
  var pin = this;
  Pin.intervals[this] = setInterval(function() {
    on = !on;
    digitalWrite(pin, on);
  }, period);
}
LED1.startFlashing(10);
LED1.startFlashing(100);
```

Chrome WebApp



Odkazy

- http://micropython.org/
- http://docs.micropython.org/en/latest/pyboard/quickref.html
- https://pycom.io/solutions/hardware/
- http://docs.micropython.org/en/latest/genrst/index.html
- http://docs.dfrobot.com.cn/upycraft/

- https://www.espruino.com/
- https://www.espruino.com/Other+Boards
- https://chrome.google.com/webstore/detail/espruino-web-ide/

Mobilní komunikace – klasické

- V rámci GSM sítě 900/1800MHz (1900)
  - CSD (Circuit Switched Data) – data plně využívající přenosové pásmo během připojení
- GPRS
  - General Packet Radio Service od GSM 1997
  - paketový systém, data se přenáší jen v případě potřeby
  - rychlost 8-20 kbit/s, podle počtu "time-slotů" pro up/down-link (celkem běžně 5, vyjímečně 6)
  - pokrytí prakticky stejné jako GSM
- EDGE (Enhanced GPRS)
  - vylepšené kódování
  - přenos max. 236.8 kbit/s, typicky do 200 kbit/s
  - pokrytí především města apod.
  - modem zpětně kompatibilní s GPRS, automaticky se přepíná
- 3G sítě – CDMA (pásmo dřívější NMT), UMTS, ...
  - menší pokrytí (především města – UMTS stále asi nejvíce O2)
  - cíleno na mobilní připojení internetu, alternativa pevné lince
- LTE (Long Term Evolution)
  - komerčně nasazen, u nás od 2013 testovací provoz, postupně pokrývání měst
  - teoretické rychlosti 172Mb/57,6Mb Dn/Up, reálně okolo 10x 3G
  - možnost hlasového provozu

Mobilní komunikace - realizace

- PCMCIA nebo ExpressCard karta
  - Nové NTB nemají PCMCIA, ExpressCard moduly stále ještě obtížněji dostupné
  - Spíše do průmyslových počítačů
- USB modul
  - Často problém s ovladači
  - Typicky spíše pro domácí než průmyslové použití
- Mobilní telefon
  - Typicky není stavěn na trvalé připojení
- Komunikační modul
  - SPI, UART (příp. USB)
  - Ovládání AT příkazy
  - Pozor na odběr, doporučen "vykrývací" aku
- Samostatný router
  - Poskytuje Ethernet připojení (nejčastěji vytváří vlastní síť pomocí NAT a DHCP)
  - Automaticky udržuje spojení
  - Existují i multi-SIM řešení pro záložní spojení
  - Možnost doprogramování funcionality (typicky Java nebo Python)

IoT a související

- IoT = Internet of Things
  - Zvyšuje se podíl zařízení s připojením k síti
    - Ethernet, levné WiFi moduly – připojení k routeru
    - BlueTooth/nRF připojení k "centrální" jednotce/routeru = Gateway
  - Ukládání dat, zpracování, analýza, ...
- Většinou staví na standardních protokolech
  - TCP/IP, Sockets, HTTP, WebServices
  - SOAP, REST, JSON
- Směřuje se k M2M (Machine-to-Machine) komunikaci
  - Zařízení si předávají data autonomně
    - Vytápění domu si "stáhne" předpověď počasí a přizpůsobí topnou křivku plánovaným venkovním teplotám
  - Nová paradigmata ukládání dat – noSQL databáze (nerelační)
    - InfluxDB – vhodná pro ukládání sérií dat (měření) bez definice struktury
    - MongoDB – dokumentově orientovaná

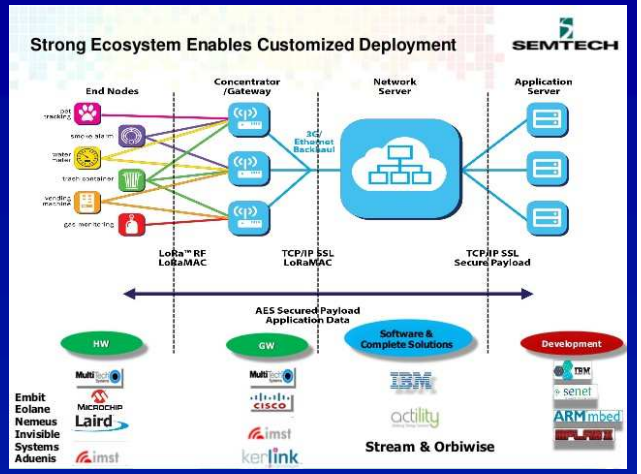
Mobilní komunikace – "malá data"

- Klasický modul pro GPRS (a lepší) komunikaci vyžaduje trvalé připojení k BTS a při komunikaci má poměrně velkou spotřebu
- Pro bateriově napájené IoT s celkem malým datovým tokem existuje (a vyvíjí se) řada nových technologií
- Při výběru nutno zohlednit:
  - Požadovaný datový tok (četnost a velikost paketů)
  - Energetickou náročnost (spotřeba)
  - Dosah
  - Pokrytí – operátoři nebo vlastní gateway
  - Další vlastnosti jako zabezpečení apod.
  - Existence vhodných čipů/modulů
- Pro běžné technologie je dostupná řada modulů i vývojových kitů + stále probíhá intenzivní vývoj nových

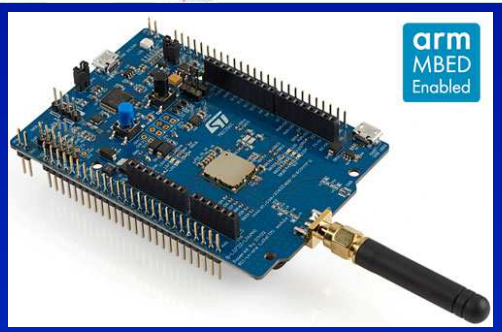
Comparison of Low-Power WAN Alternatives

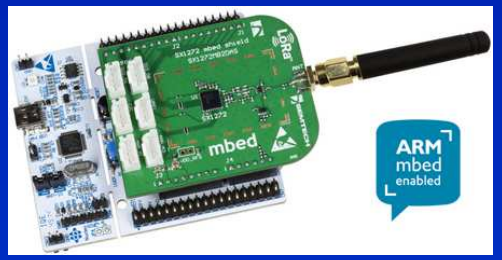
Name of Standard	Weightless			SigFox	LoRaWAN	LTE-Cat M	IEEE P802.11ah (low power WiFi)	Dash7 Alliance Protocol 1.0	Ingenu RPMA	nWave
	-W	-N	-P							
Frequency Band	TV whitespace (400-800 MHz)	Sub-GHZ ISM	Sub-GHZ ISM	868 MHz/902 MHz ISM	433/868/780/915 MHz ISM	Cellular	License-exempt bands below 1 GHz, excluding the TV White Spaces	433, 868, 915 MHz ISM/SRD	2.4 GHz ISM	Sub-GHz ISM
Channel Width	5MHz	Ultra narrow band (200Hz)	12.5 kHz	Ultra narrow band	EU: 8x125kHz, US 64x125kHz/8x125kHz, Modulation: Chirp Spread Spectrum	1.4MHz	1/2/4/8/16 MHz	25 KHz or 200 KHz	1 MHz (40 channels available)	Ultra narrow band
Range	5km (urban)	3km (urban)	2km (urban)	30-50km (rural), 3-10km (urban), 1000km LoS	2-5k (urban), 15k (rural)	2.5- 5km	Up to 1Km (outdoor)	0 – 5 km	>500 km LoS	10km (urban), 20-30km (rural)
End Node Transmit Power	17 dBm	17 dBm	17 dBm	10µW to 100 mW	EU:<+14dBm, US:<+27dBm	100 mW	Dependent on Regional Regulations (from 1 mW to 1 W)	Depending on FCC/ETSI regulations	to 20 dBm	25-100 mW
Packet Size	10 byte min.	Up to 20 bytes	10 byte min.	12 bytes	Defined by User	~100 -~1000 bytes typical	Up to 7,991 Bytes (w/o Aggregation), up to 65,535 Bytes (with Aggregation)	256 bytes max / packet	Flexible (6 bytes to 10 kbytes)	12 byte header, 2-20 byte payload
Uplink Data Rate	1 kbps to 10 Mbps	100bps	200 bps to 100 kbps	100 bps to 140 messages/day	EU: 300 bps to 50 kbps, US:900-100kbps	~200kbps	150 Kbps ~ 346.666 Mbps	9.6 kb/s, 55.55 kbps or 166.667 kb/s	AP aggregates to 624 kbps per Sector (Assumes 8 channel Access Point)	100 bps
Downlink Data Rate	1 kbps to 10 Mbps	No downlink	200 bps to 100 kbps	Max 4 messages of 8 bytes/day	EU: 300 bps to 50 kbps, US:900-100kbps	~200kbps	150 Kbps ~ 346.666 Mbps	9.6 kb/s, 55.55 kbps or 166.667 kb/s	AP aggregates to 156 kbps per Sector (Assumes 8 channel Access Point)	--
Devices per Access Point	Unlimited	Unlimited	Unlimited	1M	Uplink:>1M, Downlink:<100k	20k+	8191	NA (connectionless communication)	Up to 384,000 per sector	1M
Topology	Star	Star	Star	Star	Star on Star	Star	Star, Tree	Node-to-node, Star, Tree	Typically Star. Tree supported with an RPMA extender	Star
End node roaming allowed	Yes	Yes	Yes	Yes	Yes	Yes	Allowed by other IEEE 802.11 amendments (e.g., IEEE 802.11r)	Yes	Yes	Yes
Governing Body	Weightless SIG			Sigfox	LoRa Alliance	3GPP	IEEE 802.11 working group	Dash7 Alliance	Ingenu (formerly OnRamp)	Weightless SIG
Status	Limited deployment awaiting spectrum availability	Deployment beginning	Standard in development. Scheduled release 4Q 2015	In deployment	Spec released June 2015, in deployment	Release 13 expected 2016	Targeting 2016 release	Released May 2015	In Deployment	In Deployment

LoRa

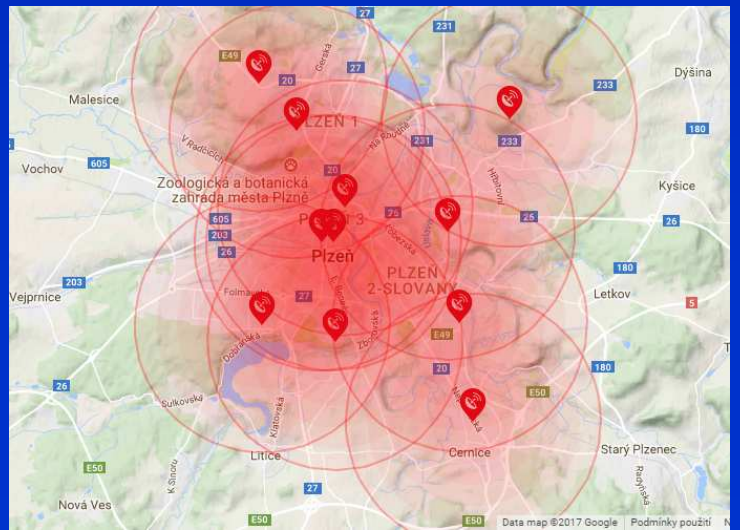


- LoRa (LOng RAnge) – implementace LoRaWAN
  - 15km volný prostor, 2-5km v zástavbě
  - 433/868/780/915MHz
  - Uplink až 50 kb/sec (v USA 100k)
  - Downlink stejný
  - Stanic na AP – UpLink > 1M, Downlink < 100k
  - Topologie – hvězda hvězd (gateway s pevným připojením)





- Moduly a kity široce dostupné
  - Např. ST B-L072Z-LRWAN1
  - Využívá moduly Semtech nebo Murata
- LoRa síť je součástí konceptu Plzeň – SmartCity
  - Podpora studentských projektů a startupů



SigFox

- Sigfox – francouzská firma
  - 30-50km volný prostor, 10km v zástavbě
  - 868/902MHz
  - Uplink 100 b/sec při 140 zprávách denně (packet 12 bajtů)
  - Downlink max. 4 zprávy denně po 8 bajtech
  - Stanic na AP = 1M
- Globální pokrytí včetně roamingu
- SigFox zaváděn v ČR od T-Mobile

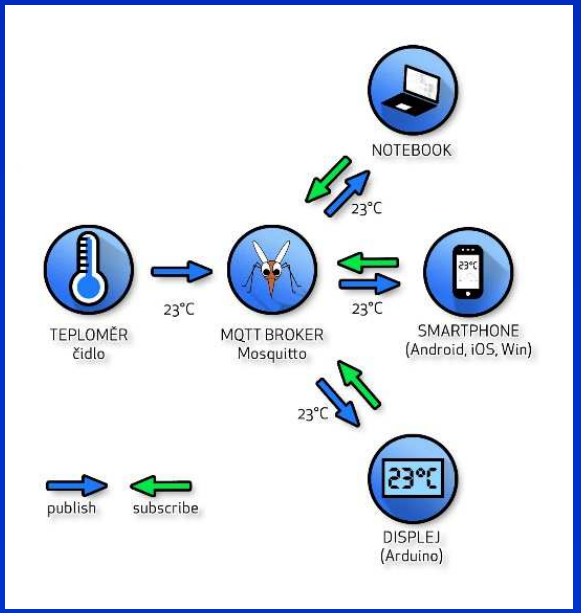
- Různé "řady" produktů podle spotřeby/výdrže/...
  - Např. Admiral Ivory = "ultralevný bezdrátový čip v hodnotě pouhých 20 centů, který dokáže skrze Sigfox přenést primitivní stavovou zprávu. Potřebuje k tomu jen zlomek energie oproti klasické komunikaci, takový čip na jedno použití a po přepočtu za pár korun by tedy mohl být součástí třeba balíku, jehož odesílatel se dozví, že jste jej rozbalili"

Další ...

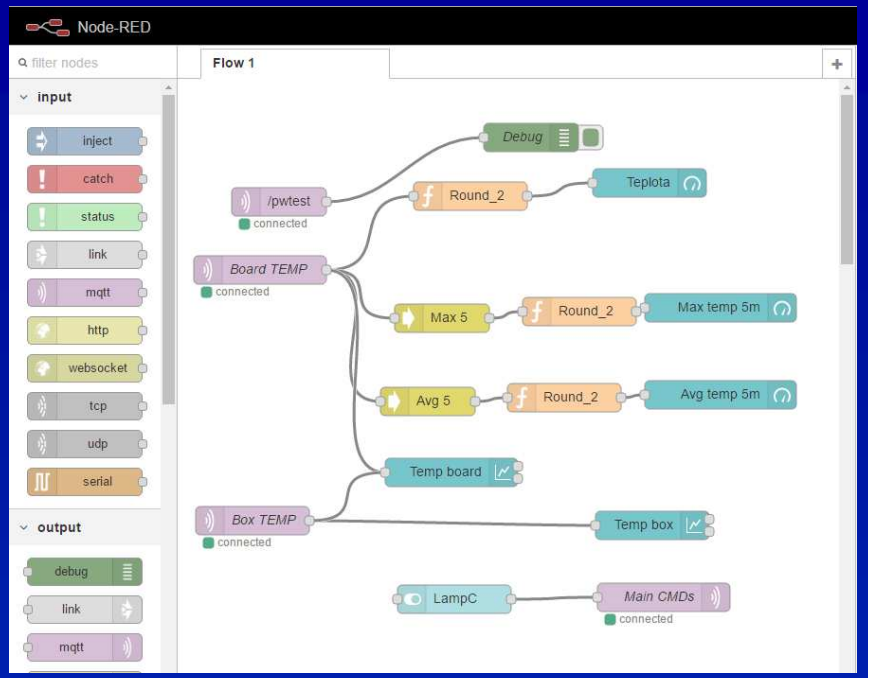
- NB-IoT
  - Využívá LTE infrastruktury a část LTE pásma
  - Relativně málo dostupných HW
    - První modul u-Blox
    - Quectel dělá řadu modulů pro GSM/LTE/... – např. BCM66 (prý) testoval Vodafone v ČR
- SubGig pásmo (868MHz v Evropě)
  - Dash-7 – dokončená specifikace
  - Např. WizziKit modul pro Nucleo-32
    - Včetně ethernet gateway

MQTT – Message Queue Telemetry Transport

- Protokol a sada nástrojů pro komunikaci mezi "zařízeními"
  - Založeno na TCP/IP, příp. SSL
  - Definována struktura zpráv
  - Doručení negarantované/garantované/"právě jednou"
  - Zpráva při odpojení, chybě, ...
  - Data v "tématech" definována textově ve stylu "stromu"
    - Např. home/inside/temperature
    - Možno použít filtrování a zástupný znak #
  - Předmětem zprávy (payload) může být jakýkoliv objekt a broker jej rozešle přesně tak, jak byl odeslán (nejjednodušší je textová reprezentace)
- Broker
  - Přijímá data a poskytuje je přihlášeným stanicím
- Senzor
  - Poskytuje data "do" brokeru
- Klient
  - Přihlásí se k odběru daného typu dat
  - Může být i senzorem (posílá data = požadavky)



MQTT
příklad řešení



- Server (Windows/Linux)
  - Musí mít veřejnou IP adresu
  - Broker např. Mosquitto
    - Využívá běžící Apache pro komunikaci
  - NodeRED (běží pod Node.JS)
    - Definice pravidel pro přeposílání a zpracování dat
    - Grafická editace i prezentace v prohlížeči
    - Obsahuje i "dashboard" s "live" zobrazením hodnot

- Senzor i klient aplikačně
  - Např. pro .Net knihovna M2Mqtt

Průmysl 4.0 a smart technologie

- Se zvyšující se automatizací a robotizací roste význam dat a komunikace
- Využívání IoT technologií
  - Chytré senzory s trvalým (dle možností) připojením
  - Datová úložiště s možností sofistikované analýzy dat
    - Nasazení algoritmů neuronových sítí apod.
- Ve finále jde o autonomní komunikaci mezi výrobními "moduly" s možností samostatné optimalizace
  - Stroj "si objedná materiál"
  - Stroj "zajistí" náhradu při přerušení výroby vlivem plánované servisní odstávky
  - ....
  - Člověk (programátor/systémový technik/atd.) je pouze na začátku = nastaví úvodní pravidla a procesy

Zajímavé a související odkazy

- http://cs.wikipedia.org/wiki/GPRS
- http://www.conel.cz/conel/index.php?newlang=czech&button=home
- http://www.nowire.cz
- http://vucako.bloguje.cz/240410_item.php
- http://mobil.idnes.cz/mob_tech.asp?c=A040613_5263509_mob_tech
- https://docs.influxdata.com/influxdb/v1.1/
- https://cs.wikipedia.org/wiki/MongoDB
- http://www.hivemq.com/
- https://mosquitto.org/
- http://mqtt.org/
- http://www.4makers.info/komari-se-zenili-aneb-neco-o-mqtt/
- https://www.rs-online.com/designspark/eleven-internet-of-things-iot-protocols-you-need-to-know-about
- http://eu.mouser.com/applications/sigfox-lora-lte
- http://www.cnx-software.com/2015/09/21/comparison-table-of-low-power-wan-standards-for-industrial-applications/
- http://www.lupa.cz/clanky/sigfox-internet-veci-bez-internetu-a-jen-pro-nektere-veci/
- http://www.slideshare.net/zahidtg/lora-introduction
- https://hackaday.com/2017/10/31/review-iot-data-logging-services-with-mqtt/
- https://www.zive.cz/bleskovky/sigfox-predstavil-unikatni-vysilac-pro-internet-veci-stoji-pouze-dvacet-centu/sc-4-a-189721/default.aspx
- https://www.itworld.com/article/3226476/internet-of-things/sigfox-shows-20-cent-iot-wireless-module.html
- https://technet.idnes.cz/iot-sigfox-world-iot-expo-internet-veci-f0f-/hardware.aspx?c=A170926_210535_hardware_kuz
- http://navody.arduino-shop.cz/navody-k-produktum/esp8266-a-thingspeak.html
- https://intenzitadopravy.plzen.eu/?hs_panel=pilsentraffic
- http://blog.st.com/wizzilab-wizzikit-da7
- https://dzone.com/articles/a-close-look-at-iot-internet-protocols
- https://technet.idnes.cz/site-lp-wan-testovani-praha-internet-veci-fgf-/
- https://www.u-blox.com/en/narrowband-iot-nb-iot
- https://iotta.cz/magaziny-o-iot/
- https://www.vodafone.cz/firmy-a-korporace/specialni-sluzby/internet-veci/nb-iot1/
- https://www.soselectronic.cz/articles/no-name/internet-of-things-3-cast-technologie-pro-bezdratovy-prenos-dat-2044
- https://www.cnx-software.com/2015/09/21/comparison-table-of-low-power-wan-standards-for-industrial-applications/
- https://iot.plzen.eu/
- http://www.st.com/content/st_com/en/products/evaluation-tools/product-evaluation-tools/mcu-eval-tools/stm32-mcu-eval-tools/stm32-mcu-discovery-kits/b-l072z-lrwan1.html
- http://simplecell.eu
- http://www.eenewseurope.com/news/narrowband-iot-shield-aris-edge
- https://www.sos.sk/gsm-umts-lte?query=lte&brand=QUECTEL&category=10577&product_show=1&application=&params=38%3ANB-IoT&param_current=38

- Články ve Sdělovací technice od cca druhé půlky roku 2016 ...

Zbývající přednášky a cvičení
Ukázky technologií aplikací

Cvičení

9. Linux pro Embedded aplikace
10. MicroPython
11. WiFi modul připojený k Nucleo-F411 – přenos dat (C aplikace)
12. Dokončení semestrální práce

Přednášky

10. tenký klient – další konstrukce ASP.NET
    - MasterPage
    - GridView, ListView, Repeater
11. Android a C# - reálné ukázky použití Xamarin
12. WPF – nové vizuální techniky pro Windows aplikace

Konec přednášek 2018/19

Další přednášky mohou být upraveny – zatím obsah z minulých let

Přednáška 10 - tenký klient ASP.NET

- MasterPage – společné části stránky
- Výpis dat – hi-level kontejnery
  - GridView – výkonná tabulka
    - Mnoho parametrů a možností
  - ListView – orientováno na seznamy
    - Možno koncipovat tabulkově i blokovými prvky
  - Repeater – nejjednodušší kontejner
    - Opakované prvky iterací seznamu
- Nastavení zdroje dat **.DataSource**, napojení na vnitřní datové struktury **.DataBind()**

Přednáška 11 – C# a Android

- Xamarin je součástí VisualStudio 17+
  - Dříve samostatné XamarinStudio příp. plugin do VS
  - Instaluje si zároveň kompletní Android SDK a Java SDK apod.

- Odlišná filozofie návrhu vizuálního vzhledu v Androidu
  - Vychází z oddělení kódu a definice vzhledu (podobně jako WPF)
  - Definice vzhledu se může lišit podle velikosti obrazovky (fyzické nebo dle rozlišení)
- Při volání funkcí Android OS dochází k překladu na volání Java-like API = výkonová ztráta
- Při využívání .Net objetků beze ztrát
  - Některé třídy jsou "výkonnejší" než nativní v Java – podle měření např. String
- Problém s dokumentací – většina příkladů/rad/řešení mimo Xamarin dokumentaci je v Java – nutné převést do C#

Přednáška 12 – ukázky na přání

- ???