Úvod

Karel Richta a kol.

katedra počítačů FEL ČVUT v Praze

Přednášky byly připraveny i s pomocí materiálů, které vyrobili Ladislav Vágner a Pavel Strnad

Programování v C++, A7B36PJC
2016, Lekce 1

https://cw.fel.cvut.cz/wiki/courses/a7b36pjc/start

Proč se učit C++?

- C++ se často používá.

Počítačová grafika

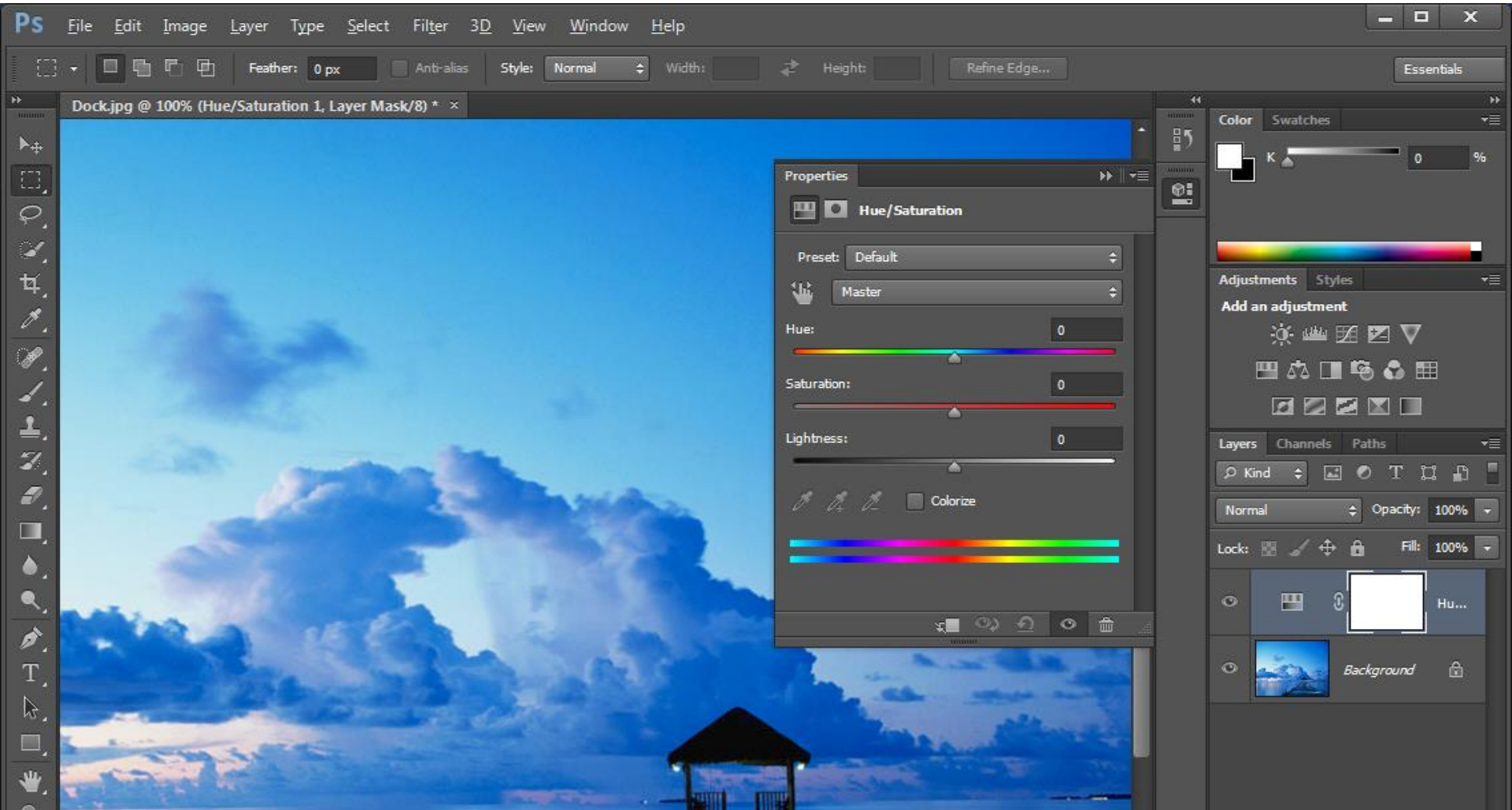
- Unreal Engine, idTech, CryEngine, Unity, ...

Analýza dat

- CERN Root

Produkční software

- CAD/CAM, editory, modeláře (Autodesk, Adobe…)

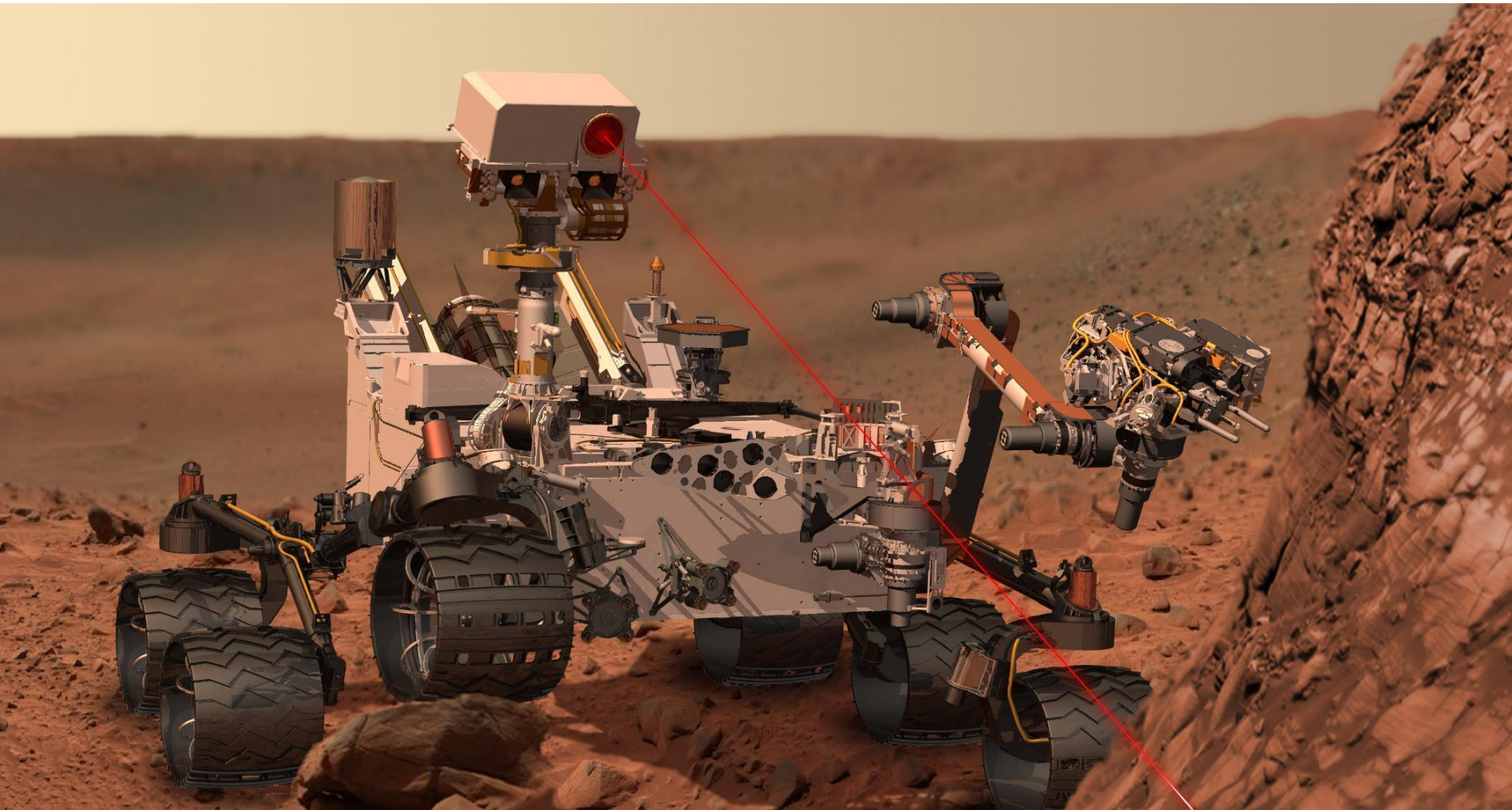


Burza a finance

- High Frequency Trading, Bloomberg

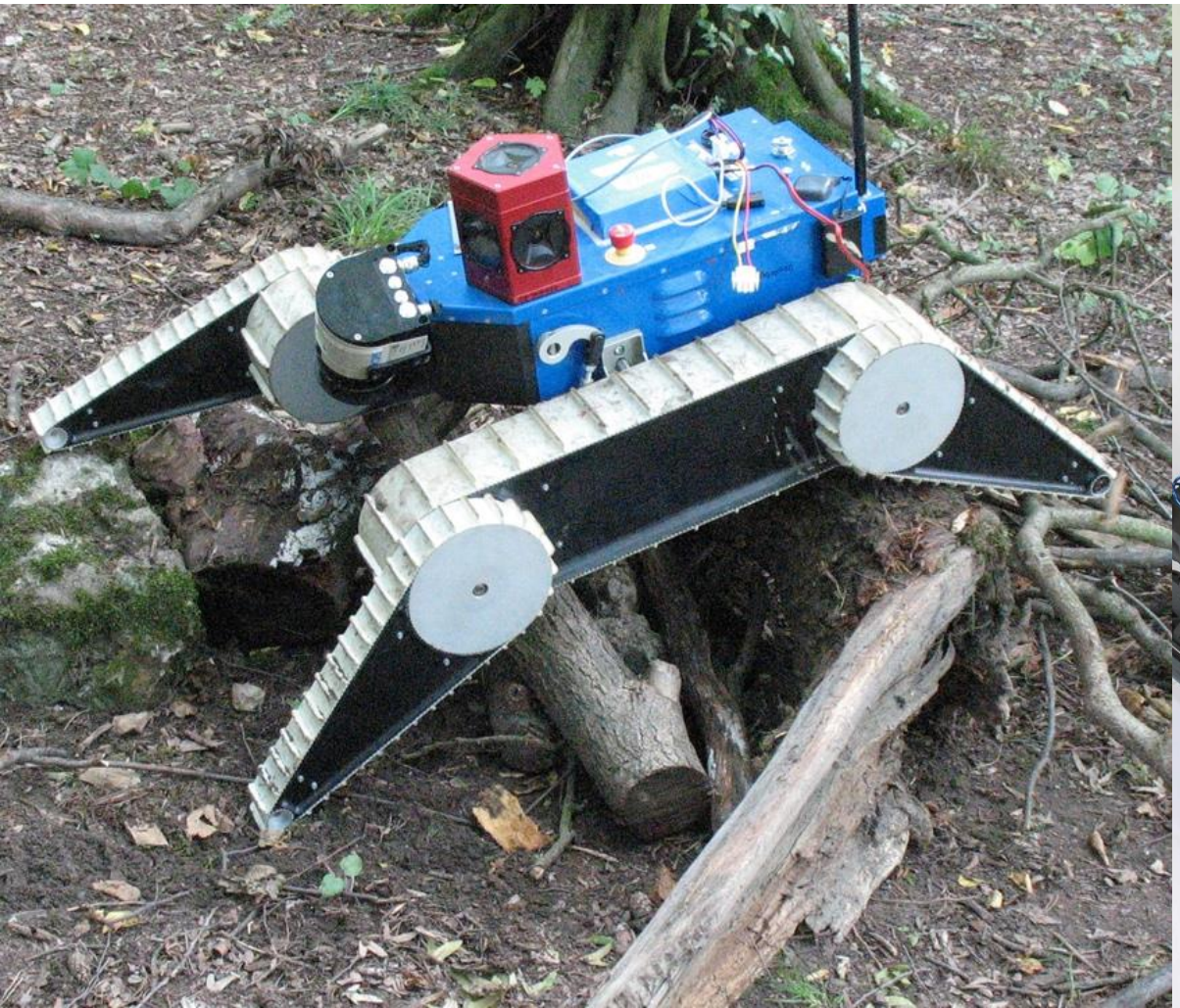
Real-time systémy

- Průmyslové technologie, na obrázku: Mars Rover



Robotika

- Robot Operating System



Zdravotnická zařízení

- Diagnostika, MRI, kardiostimulátory aj.

Přednosti jazyka C++

- Podporuje řadu programovacích stylů.
- Úzce spolupracuje s hardwarem počítače.
- Umožňuje vysokou úroveň abstrakce.
- Cena za abstrakci je nízká.
- Je zpětně kompatibilní s jazykem C.

Programovací styly v C++

- K dispozici je několik programovacích stylů:
  - procedurální programování
  - objektově orientované programování
  - funkcionální programování
  - metaprogramování se šablonami
- C++ je *imperativní* programovací jazyk, tzn. podstatou programování je tvorba *příkazů*.

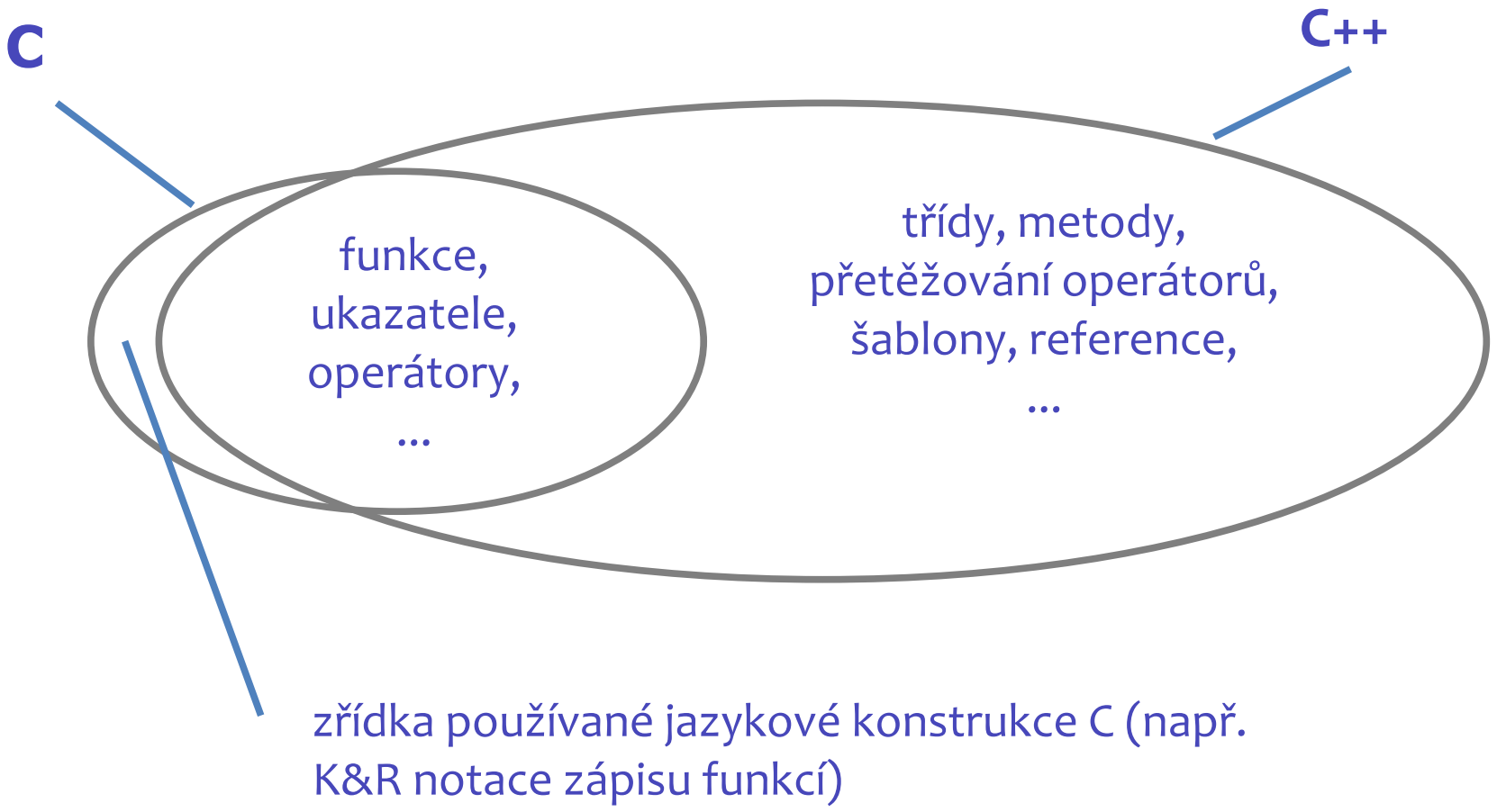
Spolupráce s hardwarem

- Přímý přístup k paměti pomocí ukazatelů.
- *Knihovny* a *ovladače* umožňují práci
  - se službami operačního systému, např. se sítěmi, souborovým systémem, správou paměti, standardním vstupem a výstupem
  - se zařízeními, např. s grafickou kartou, tiskárnou, čtečkou SD karet

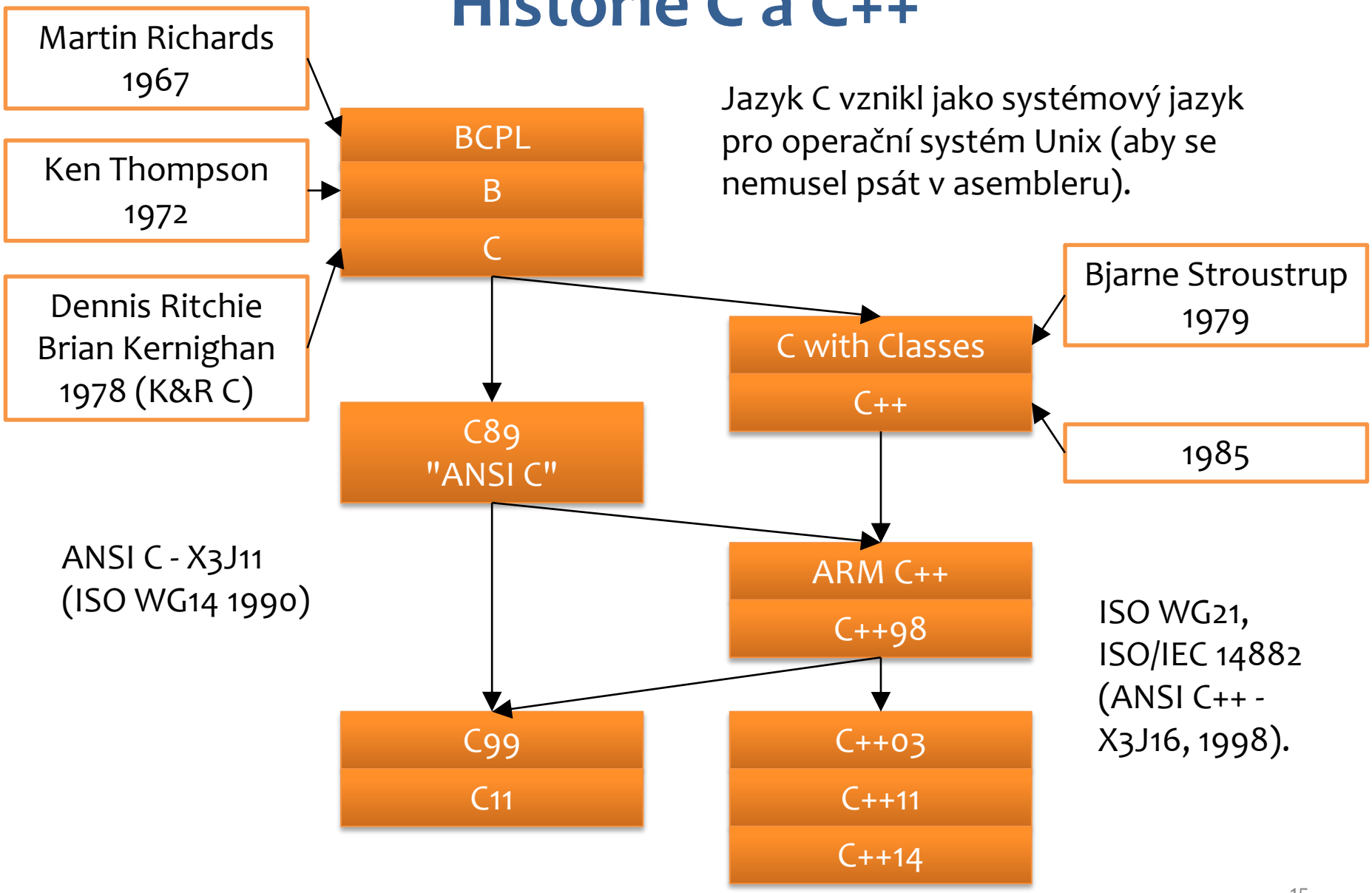
Vysoká úroveň abstrakce

- Říkáme, že C++ je jazyk *vyšší úrovně*.
- Ačkoliv zacházíme přímo s hardwarem, C++ nám poskytuje *abstrakce*, pomocí kterých lze programovat na vysoké úrovni.
- Zejména se jedná o funkce, objekty, třídy a šablony.
- Cena za použití abstrakcí je v C++ typicky nulová.

Kompatibilita C a C++



Historie C a C++



Rozdíly mezi Javou a C++

Ačkoliv syntaxe obou jazyků je velmi podobná, chování se často liší. Některé z rozdílů jsou:

- Typový systém
- Správa prostředků
- Chování při inicializaci
- Nedefinované chování
- Generické typy

Typový systém

- C++ nerozlišuje mezi primitivními typy a objekty. Například, při předávání argumentů do funkce:

Java

```
public void foo(int count, Bar b) {
    count = 3;
    b.set(4);
}
```

C++

```
void foo(int count, Bar b) {
    count = 3;
    b.set(4);
}
```

Správa prostředků

- V obou jazycích jde použít operátor new. C++ ale očekává, že je poté zavolán operátor delete.

Java

```
public void foo() {
    MyObject obj = new MyObject();
    obj.doStuff();
}
```

C++

```
void foo() {
    MyObject* obj = new MyObject();
    obj->doStuff();
}
```

Správa prostředků

- V obou jazycích jde použít operátor new. C++ ale očekává, že je poté zavolán operátor delete.

Java

```
public void foo() {
    MyObject obj = new MyObject();
    obj.doStuff();
}
```

C++

```
void foo() {
    MyObject* obj = new MyObject();
    obj->doStuff();
    // nebyl zavolán delete: ztráta paměti
}
```

Správa prostředků

- V obou jazycích jde použít operátor new. C++ ale očekává, že je poté zavolán operátor delete.

Java

```
public void foo() {
    MyObject obj = new MyObject();
    obj.doStuff();
}
```

C++

```
void foo() {
    MyObject* obj = new MyObject();
    obj->doStuff();
    delete obj;
}
```

Správa prostředků

- V obou jazycích jde použít operátor new. C++ ale očekává, že je poté zavolán operátor delete.

Java

```
public void foo() {
    MyObject obj = new MyObject();
    obj.doStuff();
}
```

C++

```
void foo() {
    MyObject obj;
    obj.doStuff();
}
```

Správa prostředků

- V obou jazycích jde použít operátor new. C++ ale očekává, že je poté zavolán operátor delete.

Java

```
public void foo() {
    MyObject obj = new MyObject();
    obj.doStuff();
}
```

C++

```
void foo() {
    MyObject obj;
    obj.doStuff();
}
```

Lepší

Správa prostředků

- Narozdíl od Javy, C++ nemá *garbage collector*, tj. neposkytuje automatickou správu prostředků.
- Výhodou pro C++ je, že prostředky jsou spravovány deterministicky:
  - buďto je paměť uvolněna v daný okamžik v kódu,
  - nebo nikdy.
- To umožňuje používat C++ v hard real-time aplikacích.

Chování při inicializaci

- Java zaručuje, že pokud vytvoříme proměnnou bez rovnítka =, proměnná bude mít danou hodnotu.

```
int i;              // hodnota i je 0
Bar b;              // reference b je null
```

- To v C++ neplatí.

```
int i;              // hodnota i je nedefinovaná
Bar* b;             // ukazatel b je nedefinovaný
```

- Proto je dobré vždy určit hodnotu proměnné:

```
int i = 0;          // hodnota i je 0
Bar* b = nullptr; // ukazatel b je nullptr
```

Nedefinované chování

- C++ standard říká, že některé situace způsobují tzv. *nedefinované chování* programu.
- Například pokud čteme z neinicializované proměnné, náš program má nedefinované chování:

```
int i;          // i je neinicializovaná proměnná
int j = i + 1; // způsobí nedefinované chování
```

- Nedefinované chování je zakázáno.
  - Kompilátor předpokládá, že program nedefinované chování nemá. To občas vede ke zvláštním situacím...
- Pokud má program nedefinované chování, může se stát *cokoliv*.

Generické typy

- V Javě je možné tvořit generické typy, ale dosadit lze pouze objekty.

```
ArrayList<int> list = new ArrayList<>();      // Chyba
ArrayList<Integer> list = new ArrayList<>(); // OK
```

- V C++ lze dosadit libovolný typ.

```
std::vector<int> list; // OK
```

Použité obrázky

- Mars Rover
  http://marsprogram.jpl.nasa.gov/msl/multimedia/images/?ImageID=3710
- Unreal Engine 4
  http://cdn.wccftech.com/wp-content/uploads/2015/03/Unreal-Engine-4-Quixel%E2%80%99s-Jungle-Environment-2.jpg
- CERN
  http://home.cern/about/experiments/cms
- Burza v New Yorku
  https://en.wikipedia.org/wiki/Stock_exchange
- Robot ROS
  http://wiki.ros.org/Robots/Erle-Rover
- Photoshop
  http://www.fullypcgames.net/2013/10/adobe-photoshop-cs6.html
- Robot NIFTI
  https://cw.felk.cvut.cz/w/_media/misc/projects/nifti/sw/adaptive_traversability/robot_obstacle_intro.jpg
- Zdravotnická zařízení
  http://getingegroup.episerverhotell.net/sv/Press/Bildgalleri/Koncern-och-affarsomraden/

Konec