Implementation Detail

LibreOffice Conference 2020

Stephan Bergmann
Red Hat, Inc.

Red Hat

C++ in LibreOffice

- Can use virtually all of C++17 now:
  - Structured bindings

    `auto [it, found] = c.insert(…);`
  - Guaranteed copy elision

    `NotCopyable factory(); auto x = factory();`
  - Class template argument detection
  - `constexpr if`
  - `[[fallthrough]]`, `[[maybe_unused]]`, `[[nodiscard]]`
  - `std::clamp`, `std::optional`, `std::string_view`, …

Going further

- C++20 is done, C++23 will come
- Opt-in `--with-latest-c++` (`-std=c++20`, `/std:c++latest`)
- Be prepared for a bumpy ride:
  - More things deprecated or removed (`std::allocator` members)
  - Synthesized `operator ==` candidates, with parameter order reversed

```
struct S { bool operator ==(S const & other); };
```

  - Compiler bugs, ...

Red Hat

Going further

- C++20 features, limited usefulness:
  - `std::span` wrapped as `o3tl::span` for now
  - `HAVE_CPP_CONSTEVAL`, `HAVE_CPP_CONSTINIT_SORTED_VECTOR`
  - No use of concepts, modules, <=>, … yet

Case Study

- Strings back in OpenOffice.org times:

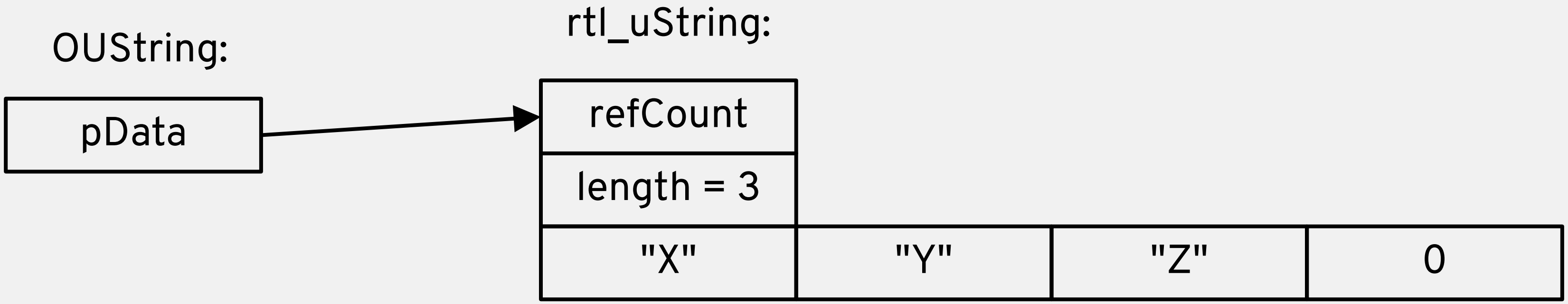
```
rtl::OUString s(RTL_CONSTASCII_USTRINGPARAM("XYZ"));
s += s2;
s += rtl::OUString(RTL_CONSTASCII_USTRINGPARAM("..."));
```

- Thanks, Luboš:

```
OUString s = "XYZ" + s2 + "...";
```

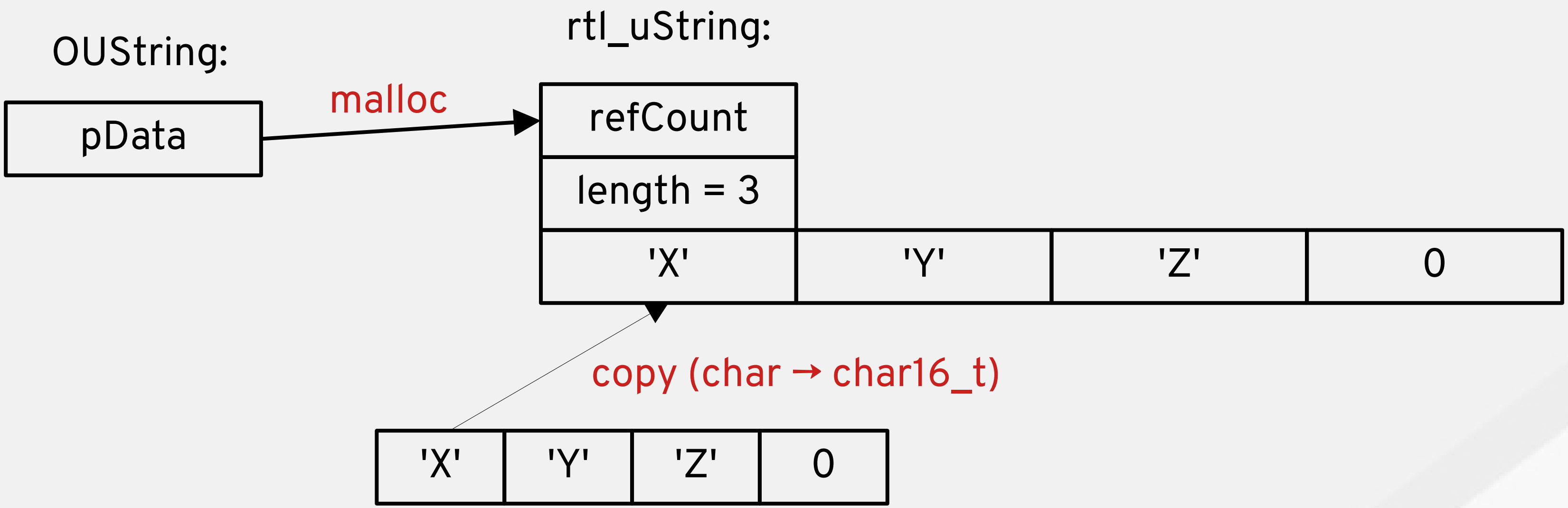
Case Study

- We are still stuck with the UNO `OUString` layout:



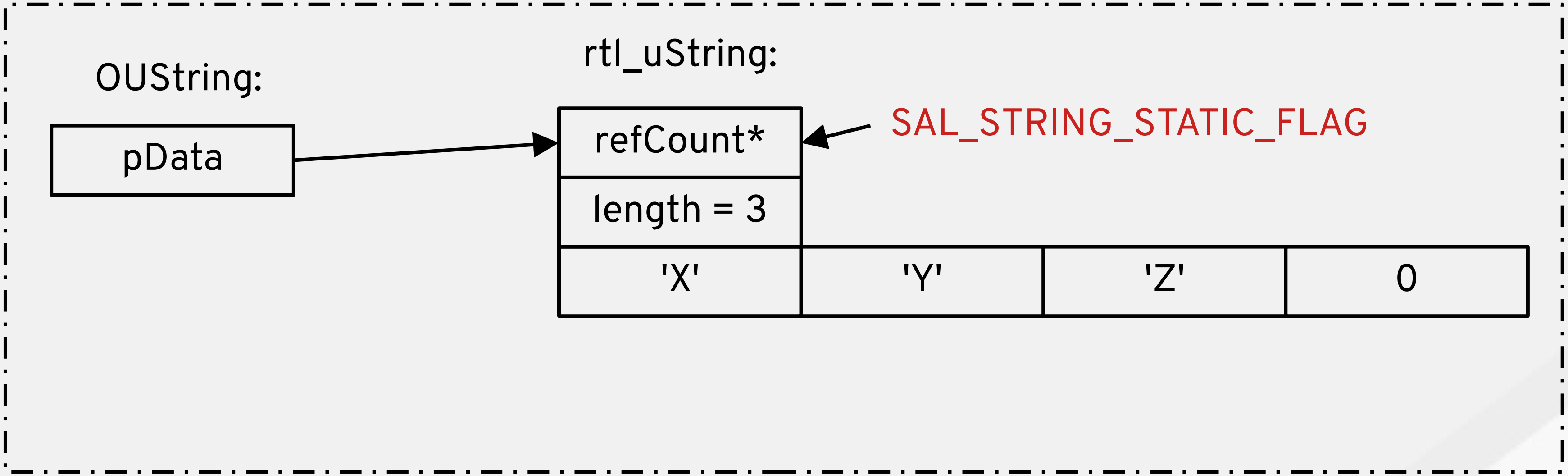
Case Study

- So `OUString s = "XYZ";` is still expensive at runtime:



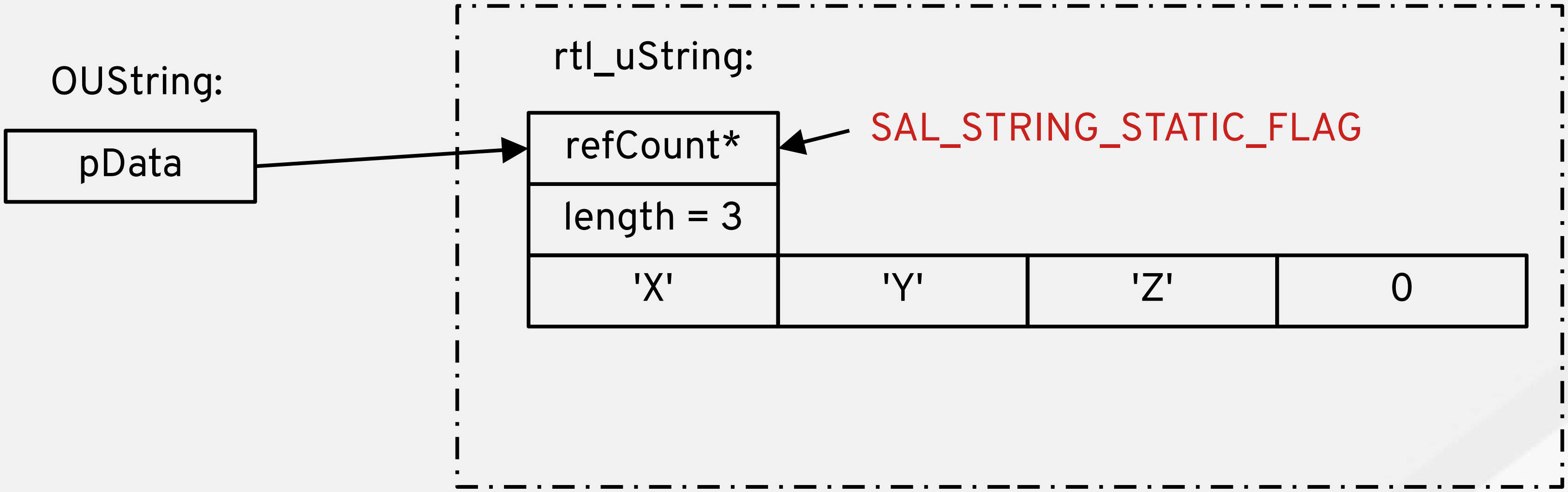
Case Study

- We want `constexpr OUString s = "XYZ";` in .rodata section:



Case Study

- `constexpr new/delete` in C++20, but must not escape from compile-time; so what about:



Case Study

- Re-purpose pre-existing `OUStringLiteral`
  - Effectively a `std::string_view`
- Often used for named constants:

```
static const OUStringLiteral Click("Click");
static const OUStringLiteral Select("Select");
static const OUStringLiteral Clear("Clear");
…
OUString s;
…
if (s == Click || s == Select) s = Clear;
```

Case Study

- Re-purpose pre-existing `OUStringLiteral`
  - Effectively a `std::u16string_view`
- Often used for named constants:

```
static const OUStringLiteral Click(u"Click");
static const OUStringLiteral Select(u"Select");
static const OUStringLiteral Clear(u"Clear");
…
OUString s;
…
if (s == Click || s == Select) s = Clear;
```

- `du -bs instdir` grew by 118.792 bytes from 1.155.636.636 to 1.155.755.428

Case Study

```
class OUStringLiteral {
  constexpr OUStringLiteral(char16_t const (&literal)[N]) {
    for (size_t i = 0; i != N; ++i) buffer[i] = literal[i];
  }
private: // same layout as rtl_uString:
  oslInterlockedCount refCount = SAL_STRING_STATIC_FLAG;
  sal_Int32 length = N - 1;
  sal_Unicode buffer[N] = {};
};

class OUString {
  OUString(OUStringLiteral const & literal):
    pData(const_cast<rtl_uString *>(
      reinterpret_cast<rtl_uString const *>(&literal))) {}
  ...
};
```

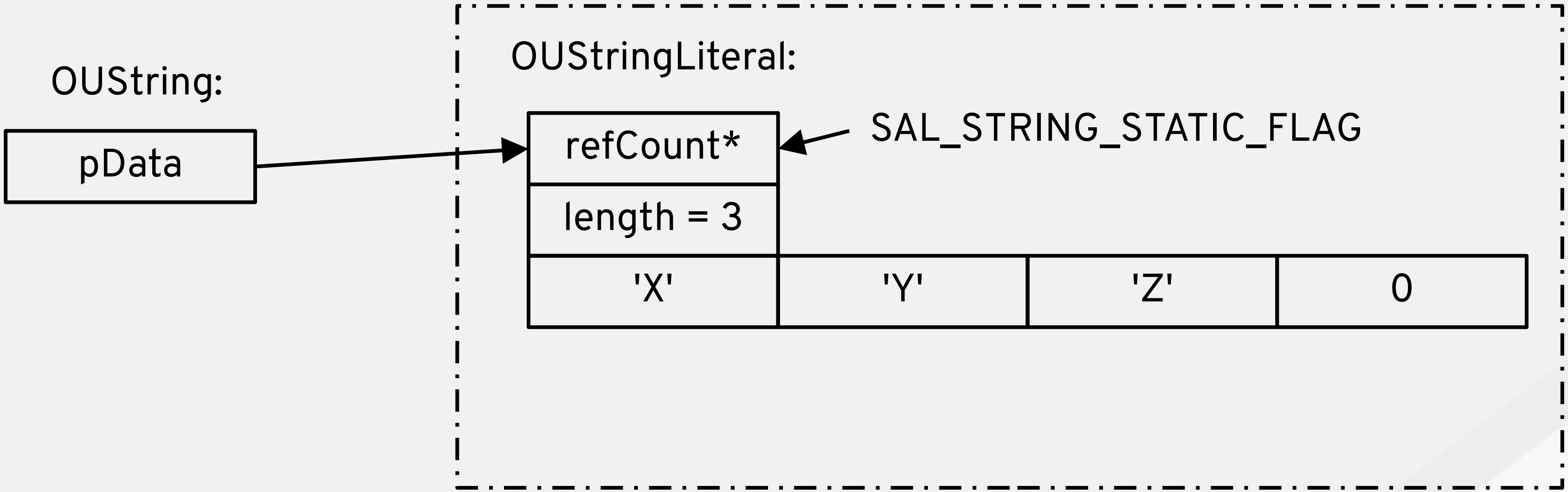
Case Study

```
template<size_t N> class OUStringLiteral {
  constexpr OUStringLiteral(char16_t const (&literal)[N]) {
    for (size_t i = 0; i != N; ++i) buffer[i] = literal[i];
  }
private: // same layout as rtl_uString:
  oslInterlockedCount refCount = SAL_STRING_STATIC_FLAG;
  sal_Int32 length = N - 1;
  sal_Unicode buffer[N] = {};
};

class OUString {
  template<size_t N> OUString(OUStringLiteral<N> const & literal):
    pData(const_cast<rtl_uString *>(
      reinterpret_cast<rtl_uString const *>(&literal))) {}
  ...
};
```

Case Study

```
static const OUStringLiteral xyz(u"XYZ"); // CTAD
OUString s = xyz;
```



Case Study

```
template<size_t N> class OUStringLiteral {
  consteval OUStringLiteral(char16_t const (&literal)[N]) {
    std::copy_n(literal, N, buffer);
  }
private: // same layout as rtl_uString:
  oslInterlockedCount refCount = SAL_STRING_STATIC_FLAG;
  sal_Int32 length = N - 1;
  sal_Unicode buffer[N]; // = {}
};
```

Case Study

```
template<size_t N> class OUStringLiteral {
  consteval OUStringLiteral(char16_t const (&literal)[N]) {
    std::copy_n(literal, N - 1, buffer);
  }
private: // same layout as rtl_uString:
  oslInterlockedCount refCount = SAL_STRING_STATIC_FLAG;
  sal_Int32 length = N - 1;
  sal_Unicode buffer[N];
};

static const OUStringLiteral xyz = u"xyz"; // fails to compile
```

Press record and then erase
I'm so fucking alone

Iggy Pop

Pop Quiz #1

```
std::string_view v1 = "XY\0Z";
v1.length();

using std::literals;
std::string_view v2 = "XY\0Z"sv;
v2.length();
```


Red Hat

Pop Quiz #2

```
...
#if defined LIBO_INTERNAL_ONLY  // can use C++17 here:
  ...
  oslInterlockedCount refCount = 0x4000'0000;
  ...
#endif
...
```


Red Hat