Supporting the “Rapi” C-language API in an R-compatible engine

Michael Sannella

msannell@tibco.com

UseR!2015

TIBCO™ Spotfire®

TIBCO™ TERR

TERR, Packages, and Rapi

- TERR: TIBCO® Enterprise Runtime for R
  - A commercial R-compatible statistics engine
  - Free Developer's Edition available
  - Commercially available for custom integration
- We want TERR to support many external packages
  - Much of the value of R lies in packages from CRAN and other repositories (Bioconductor, Github, etc.)
- Many packages are pure “R-language” packages
  - About 70% of CRAN packages are pure R-language
- Many popular packages contain C-language code accessing the R engine via R's C-language API (“Rapi”)

Example: Code Using Rapi

- R code from the CRAN package “splusTimeDate”:

```
.Call("time_to_hour_min_sec", x, timeZoneList())
```

- C code:

```
SEXP time_to_hour_min_sec(SEXP time_vec,
                          SEXP zone_list)
{
  ...
  PROTECT(ret = NEW_LIST(4));
  ...
  SET_VECTOR_ELT(ret, 0, PROTECT(NEW_INTEGER(lng)));
  ...
  hour_data = INTEGER(VECTOR_ELT(ret, 0));
  ...
  hour_data[i] = td.hour;
  ...
  UNPROTECT(7);
  return( ret );
}
```

- **`NEW_INTEGER`** macro expands to a call to the **`Rf_allocVector`** Rapi entry

TERR Rapi Support

- Ultimate goal: TERR supports loading and executing binary packages unchanged
- Fallback: Modify package sources and rebuild before loading
- If necessary, we can distribute modified packages for TERR in our TRAN repository

Rapi Shared Libraries: R.dll, etc.

- Rapi defines hundreds of library entries
  - **`ATTRIB`**, **`SET_VECTOR_ELT`**, **`R_alloc`**, **`Rf_allocVector`**, **`Rf_Protect`**, **`dcopy_`**, **`dgemm_`**, **`dlaic1_`**, etc.
  - Embedding API (used by Rstudio)
  - Global variables: **`R_GlobalEnv`**, **`R_NaInt`**, etc.
- Packages link to shared library files R.dll, Rblas.dll, etc. (libR.so, etc. on Linux) that export these Rapi entries
- To support Rapi, TERR contains R.dll, etc. libraries that forward Rapi calls to the engine
- TERR team implements Rapi entries as needed
  - Problem: Matrix 1.1-5 (released just before TERR 3.1) used **`R_compute_identical`**, which wasn't in TERR yet

TERR Rapi Support: Handles

- Observation:
  - R objects are manipulated by calling Rapi entries passing and returning R object pointers (SEXP)
- TERR's first approach: Treat SEXP as opaque handle
  - Use SEXP value as an offset in the handle table, which contains a pointer to an internal TERR object
  - Benefit: Can convert TERR objects "in-place" as needed
    - Ex: Expand TERR-specific integer sequence object to an integer vector object when calling **`INTEGER`** to access contents
  - Works well for many packages

Problem: USE_RINTERNALS

- If the C constant **`USE_RINTERNALS`** is defined, many Rapi function calls are redefined as macros directly accessing R object internals
- **`USE_RINTERNALS`** is used in many popular CRAN packages: Rcpp, Rserve, igraph, etc
- TERR workaround: Make our own versions of packages without **`USE_RINTERNALS`** defined
  - Some tweaks needed to compile:

```
STRING_ELT(x, i) = value; // change this
SET_STRING_ELT(x, i, value); // to this
```

USE_RINTERNALS More Efficient?

- Can improve efficiency of some idioms:

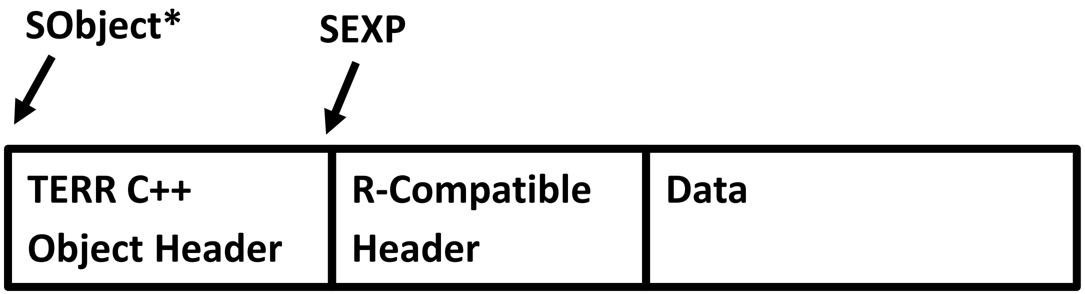
```
for (int i = 0; i<LENGTH(obj); ++i) {
   INTEGER(obj)[i] = 0;
}
```

- However, it is easy to rework code to reduce function calls:

```
int len = LENGTH(obj);
int* data = INTEGER(obj);
for (int i = 0; i<len; ++i) {
   data[i] = 0
}
```

Solution: New Object Layout

- The TERR team is currently reworking the TERR object layout to be R compatible
  - SEXP is pointer to R-compatible object
  - TERR C++ header stored before R-compatible object bytes



SObject*

SEXP

TERR C++ Object Header

R-Compatible Header

Data

New Object Layout: Issues

- Issue: List object must contain array of SEXP pointers, not TERR C++ object pointers
- Issue: TERR-only objects (like sequence objects) must be converted before exposing to Rapi code
  - Convert arguments to **`.Call`**
  - Convert value returned by Rapi entries **`Rf_eval`**, etc.
- Q: Is this worth doing?
  - Pro: Improved compatibility
  - Con: Extra complexity, object size, performance hit

Beyond USE_RINTERNALS: The data.table Package

- The data.table package exploits knowledge of engine behavior and object layout to improve performance
  - Manipulates TRUELENGTH field to reuse vectors
  - Uses TRUELENGTH field of CHARSXP as for own uses during sorting
  - Reads object bits to access string encoding quickly without Rapi function calls
- Problem: It is coding to a particular implementation of the R engine, rather than to a well-defined API

Beyond USE_RINTERNALS: The stringi Package

- Recent discovery:

```
extern "C" void R_init_stringi(DllInfo* dll)
{
  ...
  stri_set_icu_data_directory(
     (char*)*(char**)(dll) /* dll->path */);
  ...
}
```

- Uses knowledge of internal DllInfo data structure
  - May break if DllInfo structure changed
  - Would be better to get path some other way

TERR Rapi Support: Status

- TERR supports many packages with Rapi code, using handles to SEXP objects
- We are reworking TERR object layout to support packages that access object internals via **`USE_RINTERNALS`**
- We are dealing with a number of compatibility challenges