Unlocking services on z/TPF

SOA Subcommittee

Bradd Kadlecik
z/TPF Development

IBM **z/TPF**
April 11, 2016

Disclaimer

Any reference to future plans are for planning purposes only. IBM reserves the right to change those plans at its discretion. Any reliance on such a disclosure is solely at your own risk. IBM makes no commitment to provide additional information in the future.

5 Minutes	**REST**
5 Minutes	**TPF native services**
10 Minutes	**z/TPF REST interface**
5 Minutes	**Q&A**

What is REST?

- Representational State Transfer
- Better performing interface than SOAP
- Typically implemented using HTTP
- Not language dependent
- Well known by recent college grads

REST example

POST http://mytpf/loader/oldr

loads from DDN “oldr”

response provides list of loadsets loaded(LSET1,LSET2,etc)

PUT http://mytpf/loader/LSET1?action=act

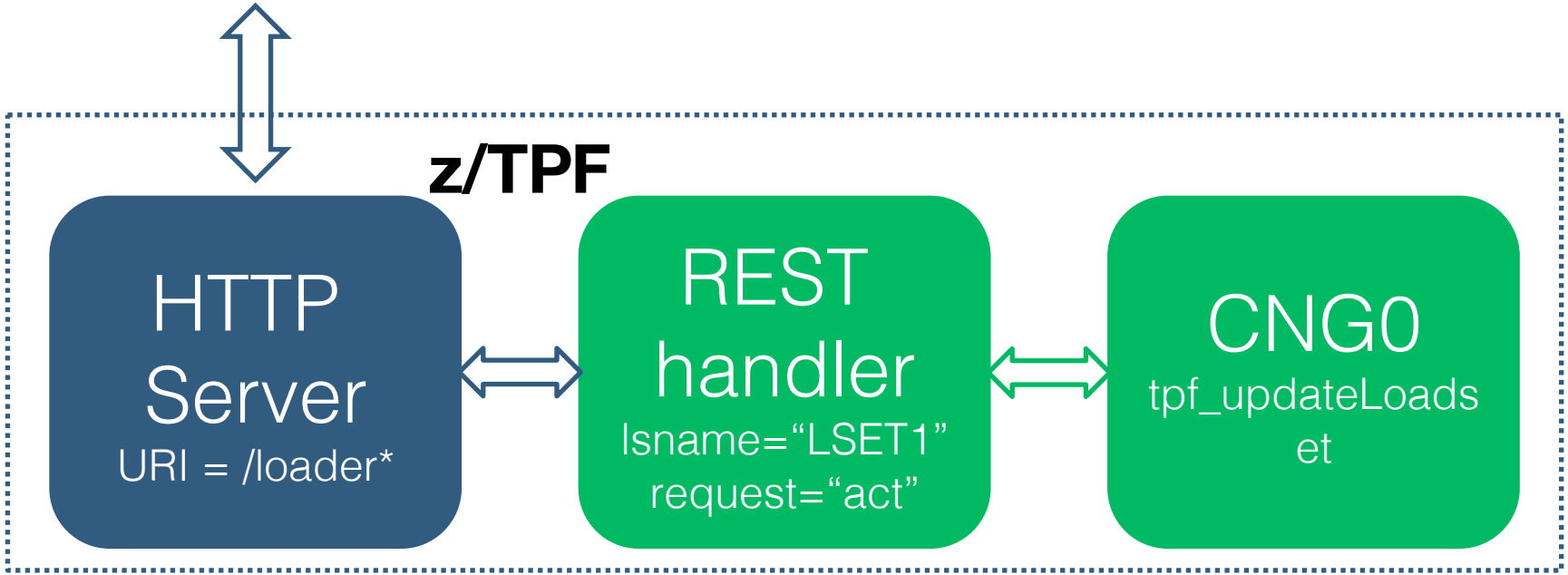
activates loadset LSET1

GET http://mytpf/loader/LSET1

response provides activation status and contents of LSET1

REST example

PUT http://mytpf/loader/LSET1?action=act



Components for REST request

- URI
- method
  - POST (Create)
  - GET (Read / Query)
  - PUT (Update)
  - DELETE (Delete)
- headers
- body (JSON,XML)

and then handle everything again for the HTTP response....

Implementing REST today requires management infrastructure and parsing.

So how can DFDL help?

XML/JSON serialization

tpf_doc_parseDocument

tpf_dfdl_serializeData

Typical Components of REST

- URI
- method
  - POST (Create)
  - GET (Read / Query)
  - PUT (Update)
  - DELETE (Delete)
- headers
- body (JSON,XML)

z/TPF can simplify the other aspects of the REST interface too

What if you just needed to code a single function and z/TPF handled the rest?

Code 1 function:

```
int tnsAPI(void *in,
           void **out);
```

and define the rest.

So how’d that work?

REST example

PUT http://mytpf/loader/LSET1?action=act

activates loadset LSET1

Code TPF native service API

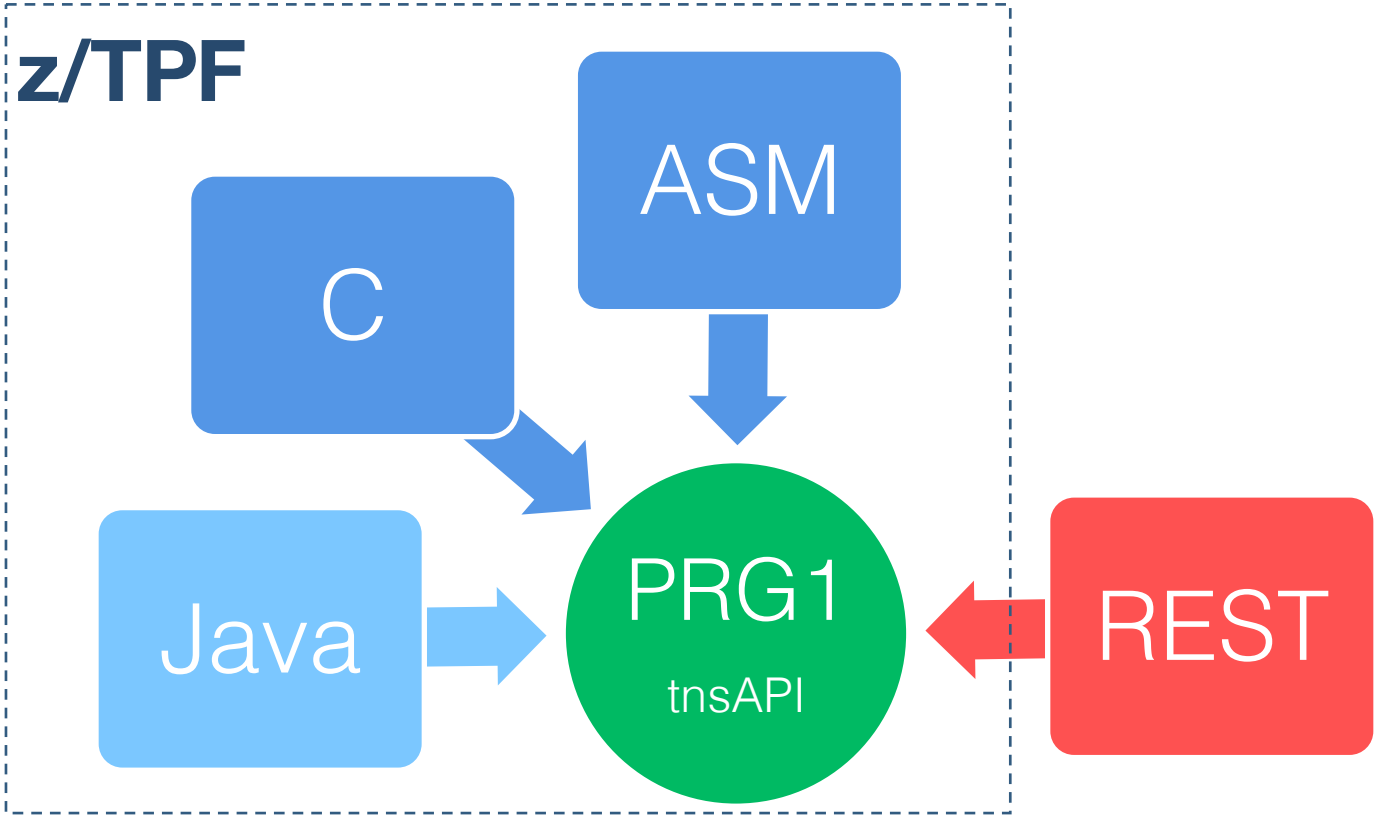
Update program CNG0

New file celtns.c, to contain the following function:

```
int tpf_updateLoadset(struct lsetinfo *in,  void **out);

struct lsetinfo {
   unsigned char lsname[8];
   unsigned char request[3];
};
```

TPF Native Service Integration

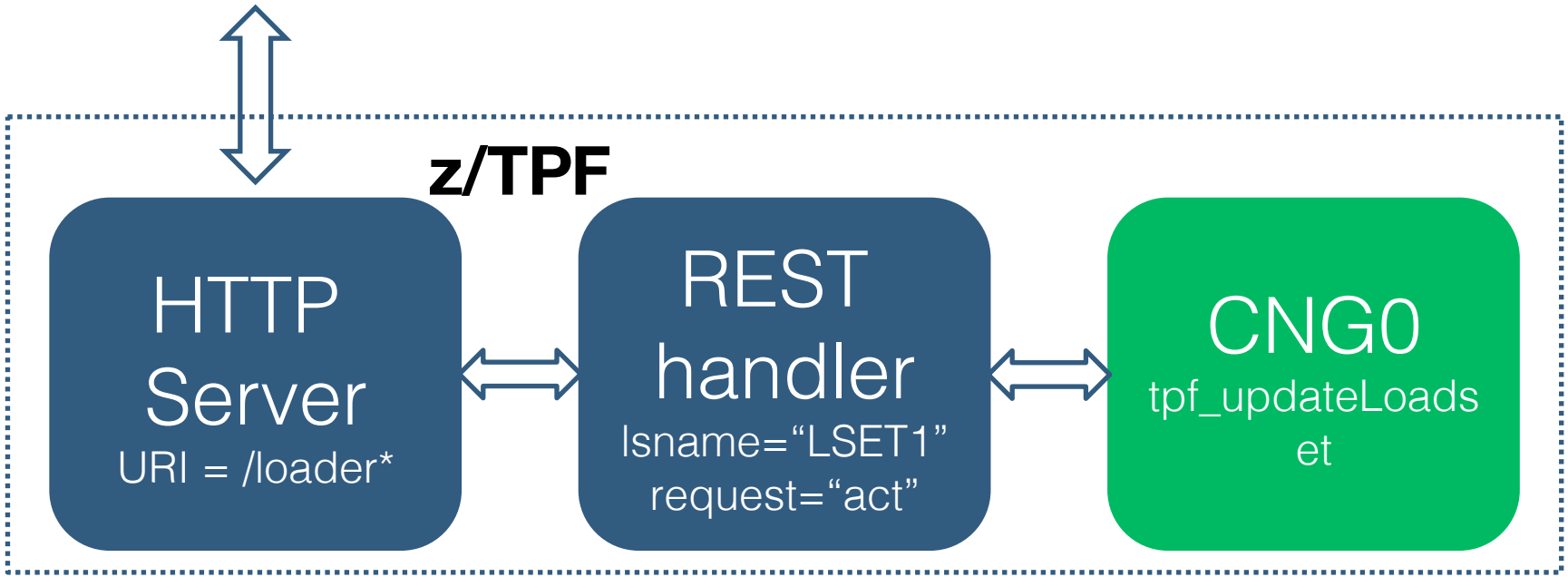


Configure z/TPF for REST

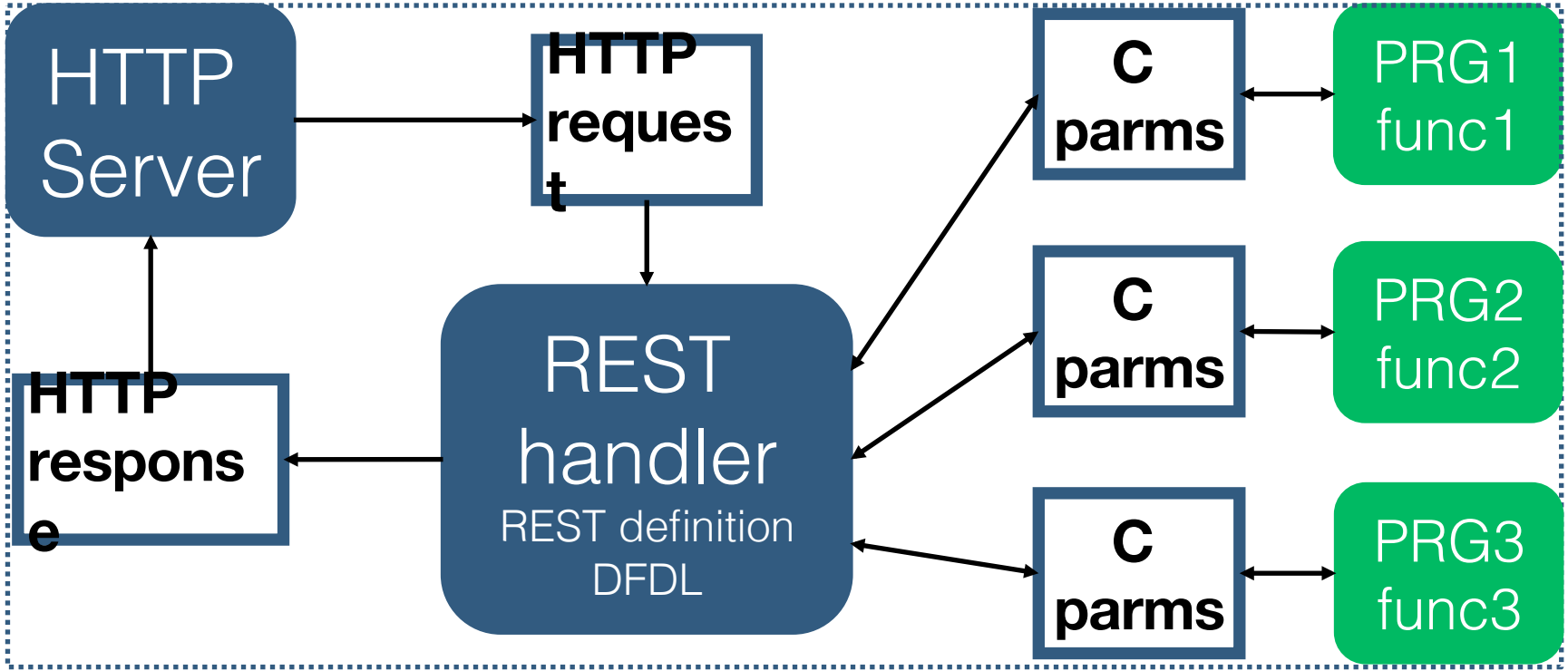
- Run **maketpf CNG0 celtns.o dfdl** to generate **lsetinfo.gen.dfdl.xsd** for lsetinfo struct.
- Define REST interface in file **loader.swag.json**.
- Load **CNG0**, **lsetinfo.gen.dfdl.xsd**, and **loader.swag.json** to z/TPF and deploy loader.swag.json through common deployment.
- Update z/TPF HTTP server URL-program mapping file to register the service.

REST example

PUT http://mytpf/loader/LSET1?action=act



REST infrastructure



z/TPF REST interface

- Provides centralized REST infrastructure for service definition and management.
- Exposes services via REST using configuration data instead of having to write a bunch of code, allowing faster deployment.
- Enables integration of services in a distributed system to allow the services to be called either remotely via REST or locally via Java.
- Allows for future enablement of other REST API services.

Thank you!

Questions or comments?

Trademarks

- IBM, the IBM logo, ibm.com and Rational are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Notes

- Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.
- All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.
- This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.
- All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.
- Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.
- Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.
- This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.