Narayana: Java library for transaction processing

DevConf CZ 2021

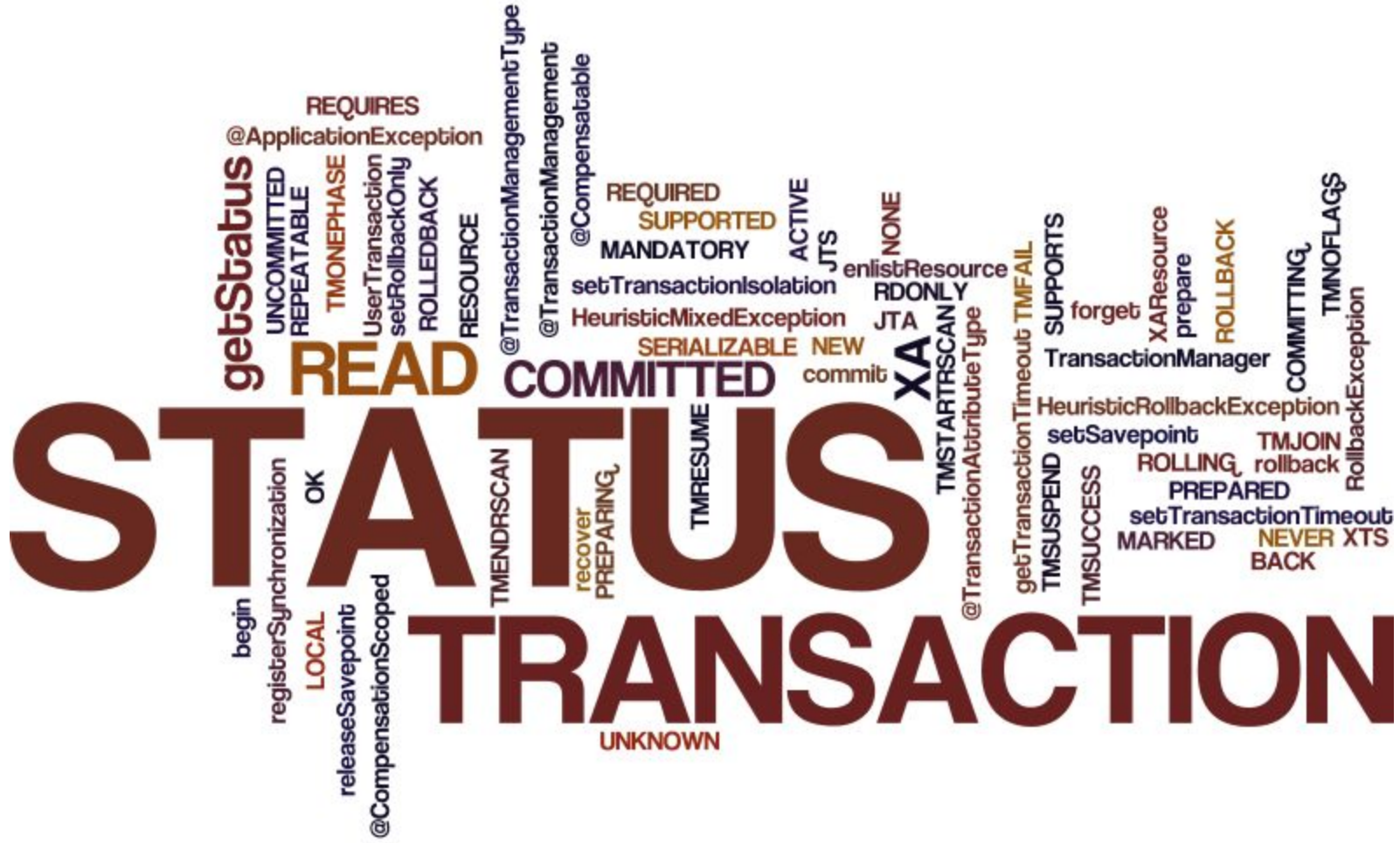
Ondra Chaloupka

Red Hat

Goal: Overview of Narayana capabilities

Schedule: Presenting each module in few words

Red Hat


STATUS
TRANSACTION
READ
COMMITTED
getStatus
REQUIRES
@ApplicationException
UNCOMMITTED
REPEATABLE
TMONEPHASE
UserTransaction
setRollbackOnly
ROLLEDBACK
RESOURCE
@TransactionManagementType
@TransactionManagement
@Compensatable
REQUIRED
SUPPORTED
MANDATORY
ACTIVE
JTS
NONE
enlistResource
RDONLY
setTransactionIsolation
HeuristicMixedException
JTA
SERIALIZABLE
NEW
commit
XA
TMSTARTRSCAN
@TransactionAttributeType
getTransactionTimeout
TMFAIL
SUPPORTS
forget
XAResource
prepare
ROLLBACK
TransactionManager
COMMITTING
TMNOFLAGS
RollbackException
HeuristicRollbackException
setSavepoint
TMJOIN
ROLLING
rollback
PREPARED
setTransactionTimeout
MARKED
NEVER
XTS
BACK
TMSUSPEND
TMSUCCESS
TMRESUME
TMENDRSCAN
recover
PREPARING
OK
registerSynchronization
begin
LOCAL
releaseSavepoint
@CompensationScoped
UNKNOWN

Transaction
=
unit of work

Transaction “model”
=
provides guarantees

Transaction library
=
makes model working

- Java library and framework for transaction processing
  - https://narayana.io
  - https://github.com/jbosstm/narayana
  - https://groups.google.com/forum/#!forum/narayana-users
- Integrated in various projects
  - WildFly application runtime
  - Quarkus application framework
  - Apache Tomcat server
  - Apache Camel
  - Spring framework


World of Narayana

Transaction Manager

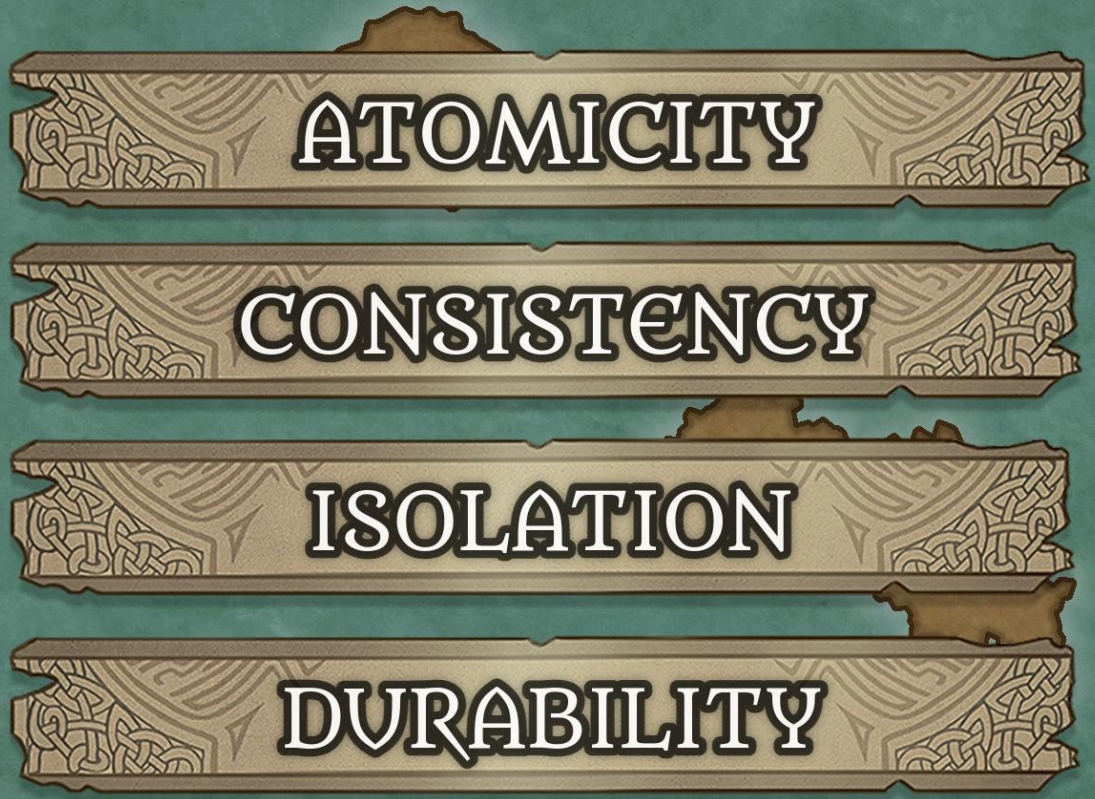
Thief task

Warrior task

Database
cloak
resource

Messaging
picklock
resource


Unit of work



Transactional guarantees


JTS transactions
JTA transactions
Software Transaction Memory
XTS
Long Running Action
REST - AT
World of Narayana
Transaction library


JTS transactions
JTA transactions
Software Transaction Memory
XTS
Long Running Action
REST - AT

- OTS (Object Transaction Service) standard
  - part of the CORBA services
- OTS defines operations as methods on objects
  - using IIOP as the communication protocol
- JTS (Java Transaction Service)
  - Specification for transactional interoperability between EJB containers based on CORBA, OTS and JTA

OTS Naming service

Transaction Manager

Thief service

Warrior service

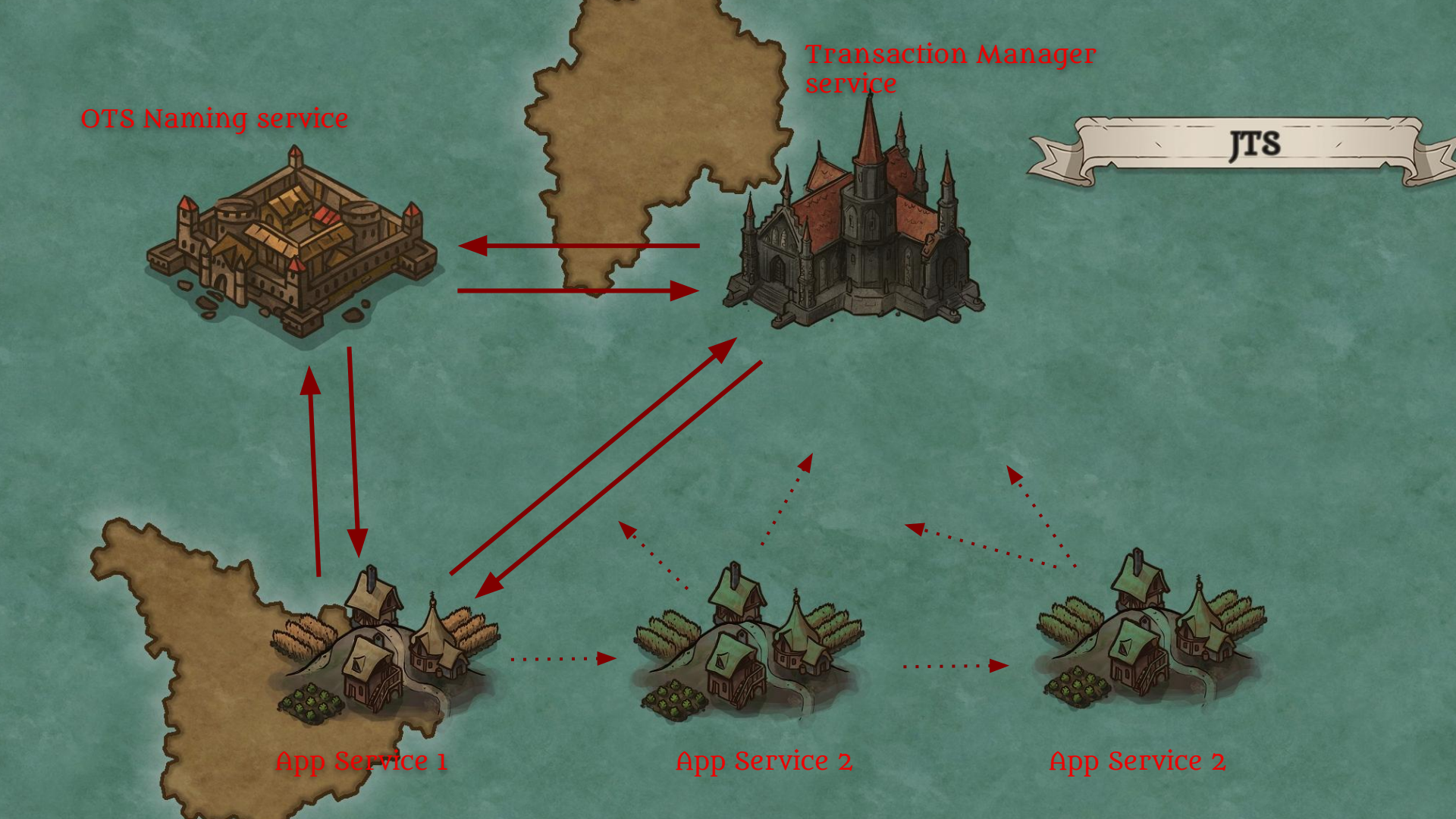
OTD idl

```
module demo {
                /*
Implicit transaction propogation characteristics
                 */
                interface Hello:
CosTransactions::TransactionalObject
                {
                    /*
                     Under transactional control.
                     */
                    void print_hello();
                };
};
```

protobuf idl

```
service Foo {
       rpc Bar(FooRequest) returns(FooResponse);
}
```

- framework to build on top of
- IIOP messages that services communicate with each other
- “ancient” with respect to current software development


Transaction Manager service
OTS Naming service
JTS
App Service 1
App Service 2
App Service 2


JTS transactions
JTA transactions
XTS
Software Transaction Memory
Long Running Action
REST - AT

- JTA (Java Transaction API)
  - distributed transactions to be done across multiple X/Open XA resources
- Integrated with EJB and CDI in application runtimes
  - Mostly transparent to developers

Application Runtime

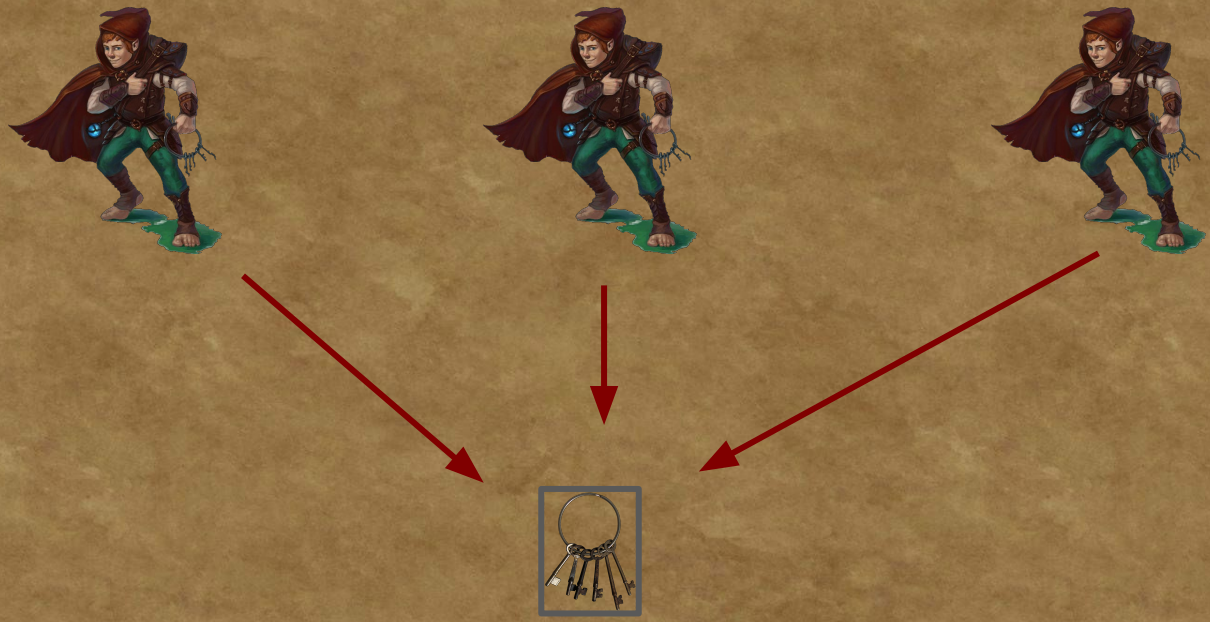
Transaction Manager

Warrior component

Thief component

- Java API to work with XA transactions / 2-phase commit protocol
- heavily used
- integrated to various runtime
- transparent for developer


JTS transactions
JTA transactions
Software Transaction Memory
XTS
Long Running Action
REST - AT



- STM (Software Transactional Memory)

- A concurrency models which uses shared memory
- An alternative to the lock-based synchronization approach
- Grouping memory operations to run them atomically

```
int x = 0, y = 0, z = 0;
```

```
void first {
   synchronized(this) {
       x = x + z;
   }
}
```

```
void second {
   synchronized(this) {
       x = x + z;
       y = y + 1;
   }
}
```

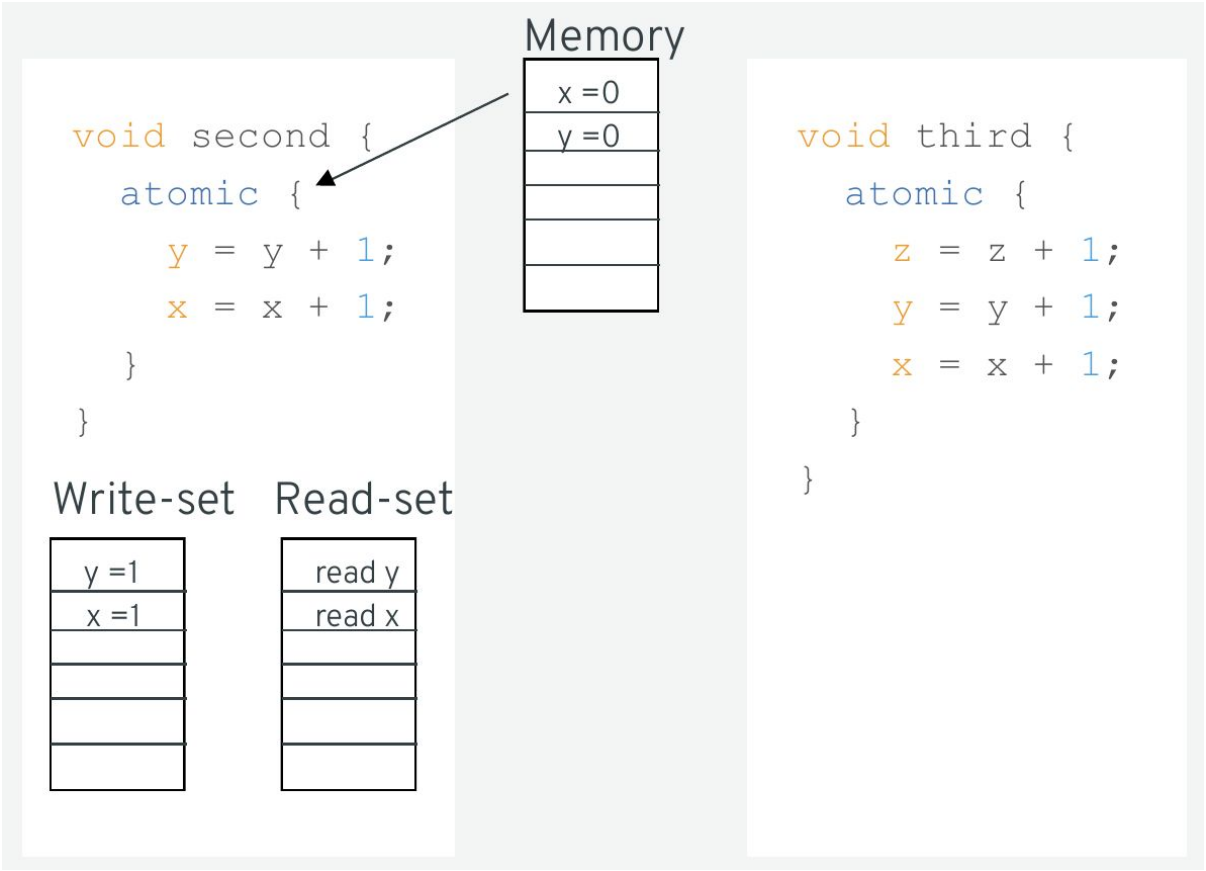
```
void third {
   synchronized(this) {
       x = x + 1;
       y = y + 1;
       z = z + 1;
   }
}
```

```
int x = 0, y = 0, z = 0;
```

```
void first {
   atomic (this) {
       x = x + z;
   }
}
```

```
void second {
   atomic(this) {
       x = x + z;
       y = y + 1;
   }
}
```

```
void third {
   atomic (this) {
       x = x + 1;
       y = y + 1;
       z = z + 1;
   }
}
```


Memory
x =0
y =0
void second {
atomic {
y = y + 1;
x = x + 1;
}
}
void third {
atomic {
z = z + 1;
y = y + 1;
x = x + 1;
}
}
Write-set
Read-set
y =1
x =1
read y
read x

- optimistic locking approach for in-VM mutual exclusion

JTS transactions

JTA transactions

Long Running Action

XTS

Software Transaction Memory

REST - AT

- LRA (Long Running Actions)
  - MicroProfile specification proposal https://github.com/eclipse/microprofile-lra
- REST services communicates with transaction manager
- Saga pattern
  - Not ACID - only ACD (atomicity, consistency, durability)
  - Complete/Compensate callbacks

- Much more details at DevConf 2021 at session A different flavor of the distributed transaction
  - Saturday, February 20, 9:45 am

Transaction Manager
/ LRA Coordinator
(REST API)

Thief service

Warrior service

- Annotations to declare the processing
- Saga pattern
- Business logic takes responsibility for compensation
- Services communicates with HTTP (JAX-RS)

- Attend A different flavor of the distributed transaction
  - Saturday, February 20, 9:45 am


JTS transactions
JTA transactions
XTS
Software Transaction Memory
Long Running Action
REST - AT

- XTS (XML Transaction Service)
  - Transaction specification for JAX-WS (SOAP based web services)
- WS-AT (WS Atomic Transaction)
  - Atomic transactions with ACID guarantees
  - Capability of bridging JTA transactions to WS-AT
- WS-BA (WS Business Activity)
  - Saga based processing

- With WS-AT the JTA transaction may be spanned over multiple services

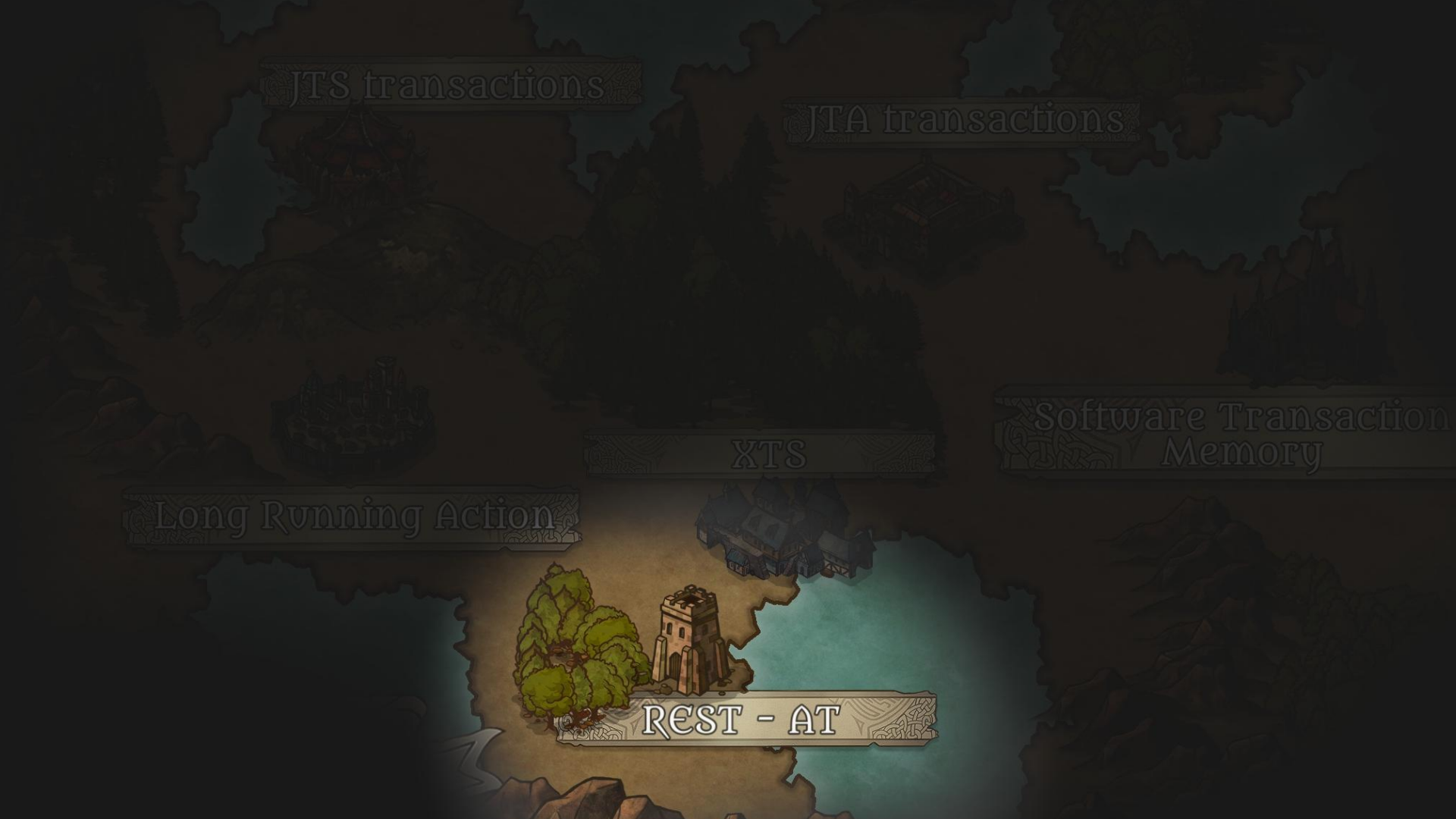
Transaction
Manager

Thief
component

Transaction
Manager

Warrior
component

- Transaction over JAX-WS - SOAP based web services
- WS-AT for fully ACID / JTA transactions
- WS-BA for saga based approach


JTS transactions
JTA transactions
Software Transaction Memory
XTS
Long Running Action
REST - AT

- REST-AT (Rest Atomic Transactions)
  - Narayana proprietary protocol for handling transaction context over Restful Web Services
- Spanning JTA transactions over HTTP calls

- Narayana modules
  - JTS
  - JTA
  - STM
  - LRA
  - XTS
  - RTS
- Slides and sources at
  - https://github.com/ochaloup/devconf2021-narayana-journey