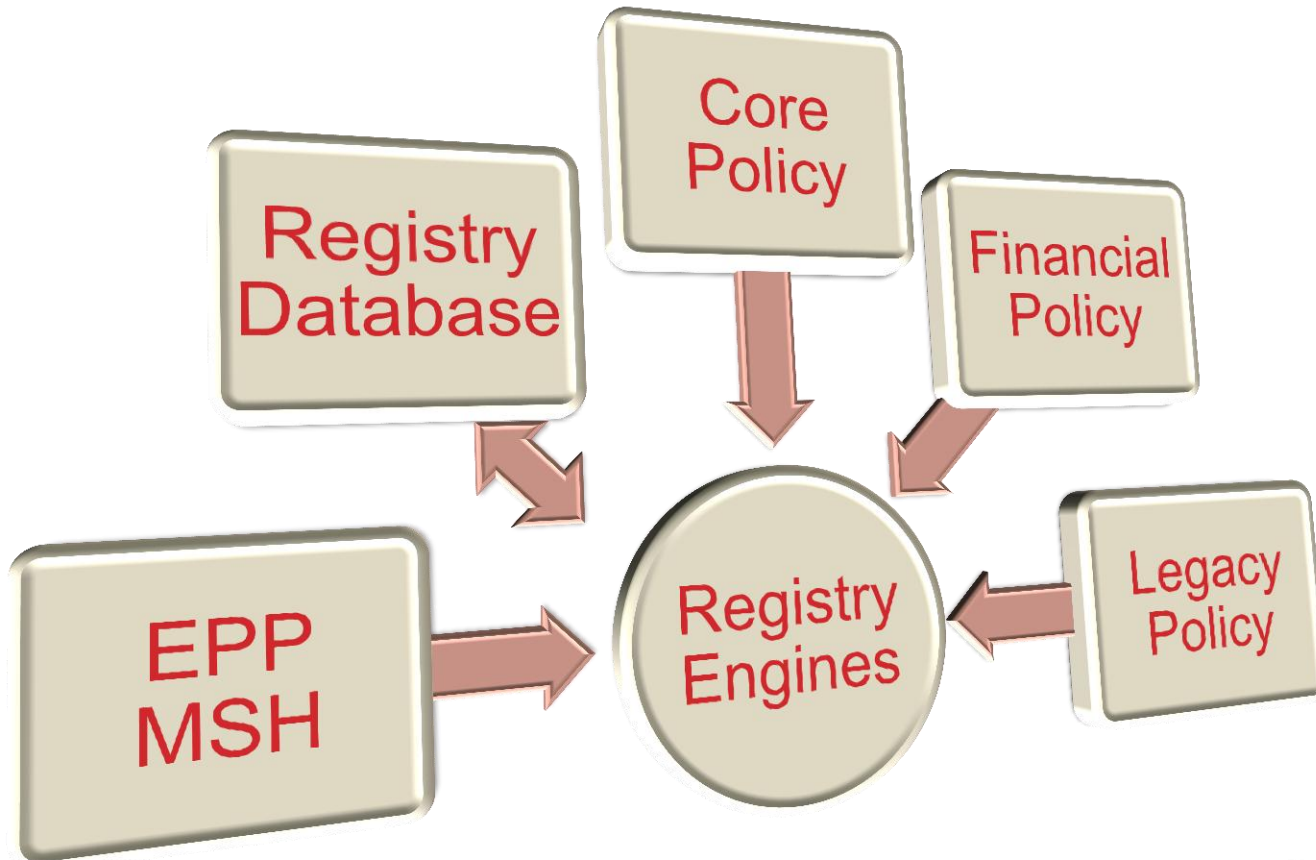


# African Top Level Registry

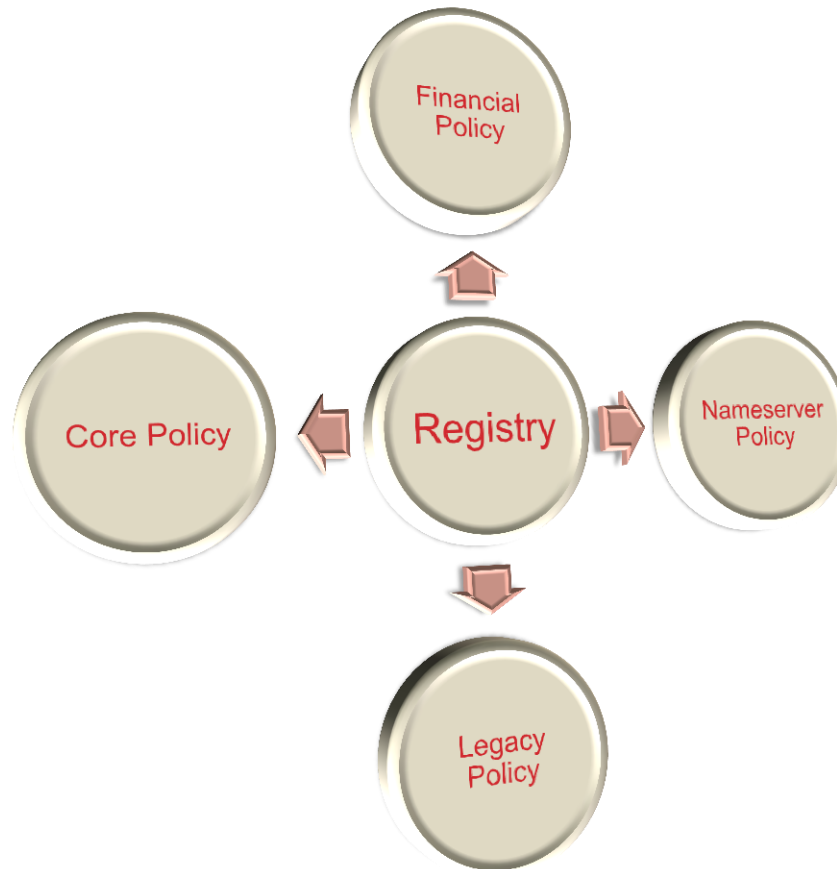


**AN AFRICAN REGISTRY SOLUTION**

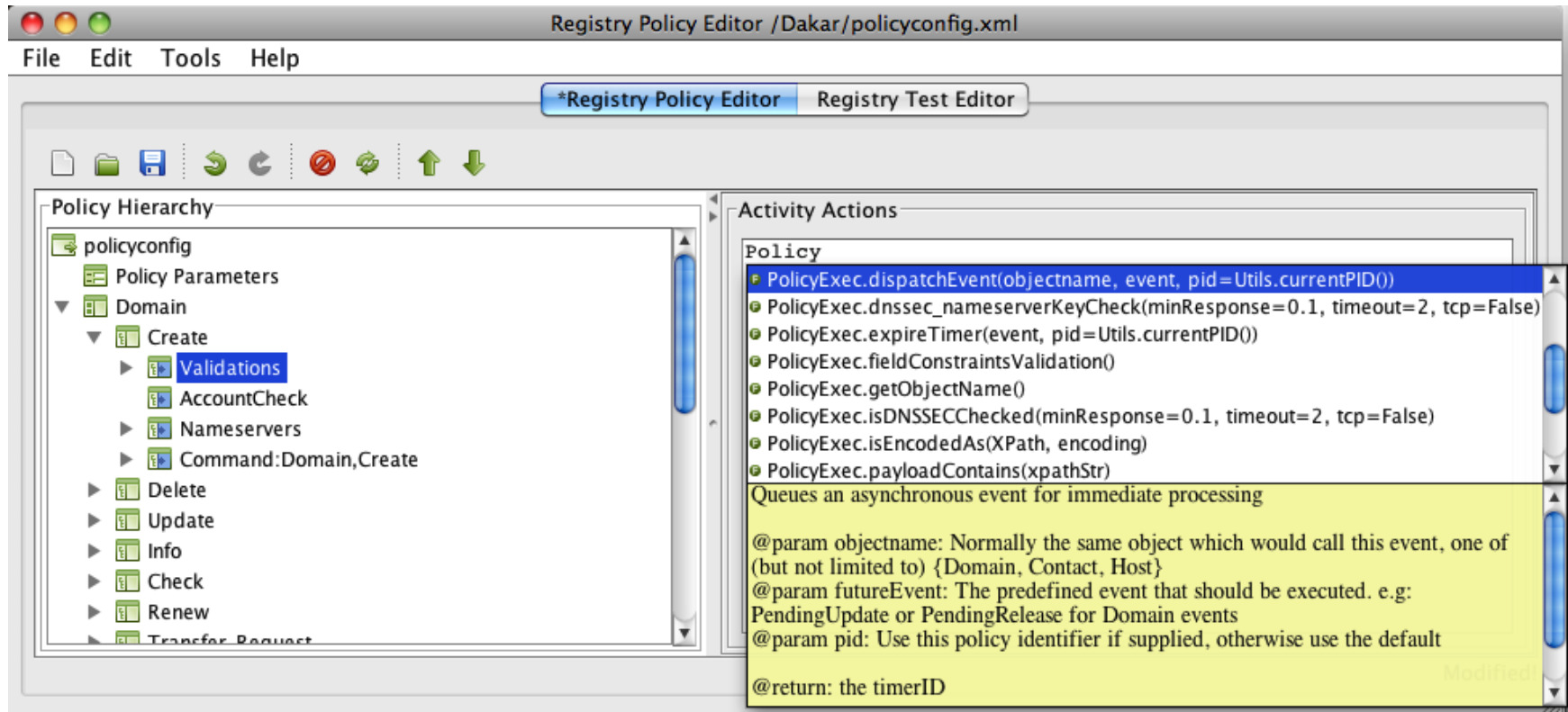
# Overview



# Policy Delegation



# Policy Creation

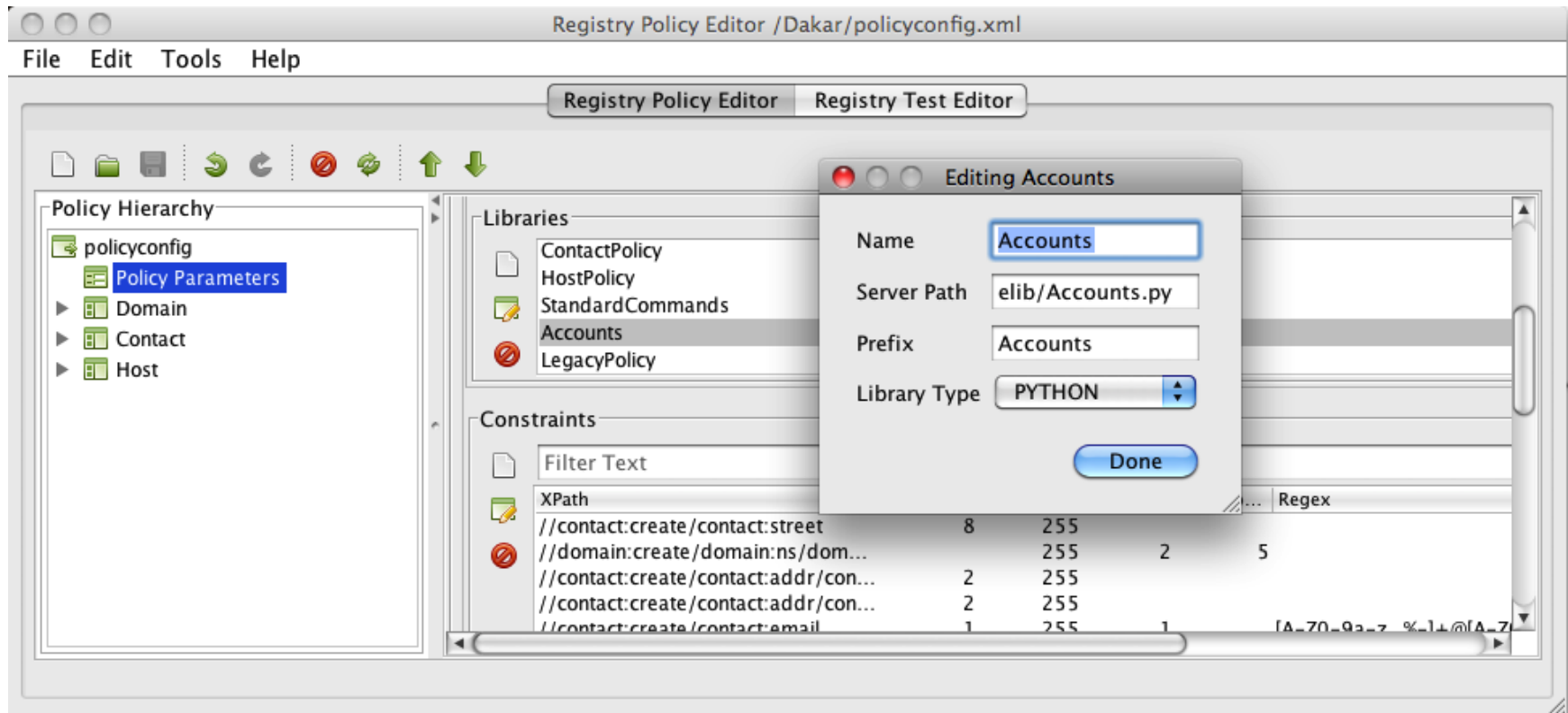


The screenshot shows the Registry Policy Editor application window titled "Registry Policy Editor /Dakar/policyconfig.xml". The interface includes a menu bar (File, Edit, Tools, Help) and a toolbar with various icons. The main workspace is divided into two panes:

- Policy Hierarchy:** A tree view showing the configuration structure. The "Domain" folder is expanded, and the "Validations" sub-item under "Create" is selected.
- Activity Actions:** A list of actions for the selected "Validations" item. The first action, "PolicyExec.dispatchEvent(objectname, event, pid=Utils.currentPID())", is highlighted. A yellow tooltip box provides details for this action:
  - Policy:** PolicyExec.dispatchEvent(objectname, event, pid=Utils.currentPID())
  - Description:** Queues an asynchronous event for immediate processing
  - Parameters:**
    - @param objectname: Normally the same object which would call this event, one of (but not limited to) {Domain, Contact, Host}
    - @param futureEvent: The predefined event that should be executed. e.g: PendingUpdate or PendingRelease for Domain events
    - @param pid: Use this policy identifier if supplied, otherwise use the default
  - Return:** @return: the timerID



# Policy Extensions



Filter Text	...	Regex
//contact:create/contact:street	8	255
//domain:create/domain:ns/dom...		255 2 5
//contact:create/contact:addr/con...	2	255
//contact:create/contact:addr/con...	2	255
//contact:create/contact:email	1	255 1 [A-Z0-9a-z_%-]+@[A-Z



# Policy Implementation

Registry Policy Editor /Dakar/policyconfig.xml

Registry Policy Editor Registry Test Editor

Policy Hierarchy

- Domain
  - Create
  - Delete
    - ConstraintsCheck
    - DomainStatusCheck
    - DependencyCheck
    - DomainOwnershipCheck
    - Command:Domain,Delete,DryRun
      - GracePeriodCheck
        - DomainPolicy.isInGracePeriod(PolicyPeriod)
          - InGracePeriod
            - DomainState:PendingRelease
            - Timer:Domain,PendingGracePeriodSuspension

Activity Actions

```
#Pol.Ver6: 6.3.3.2 On expiry of the remainder of  
# the grace period, the domain name will automatically be  
# deleted (removed from the registry database), and revert  
# back into the pool of available domain names.  
PolicyExec.startTimer(  
    "Domain",  
    "PendingGracePeriodSuspension",  
    PolicyPeriod)
```



# Policy XML

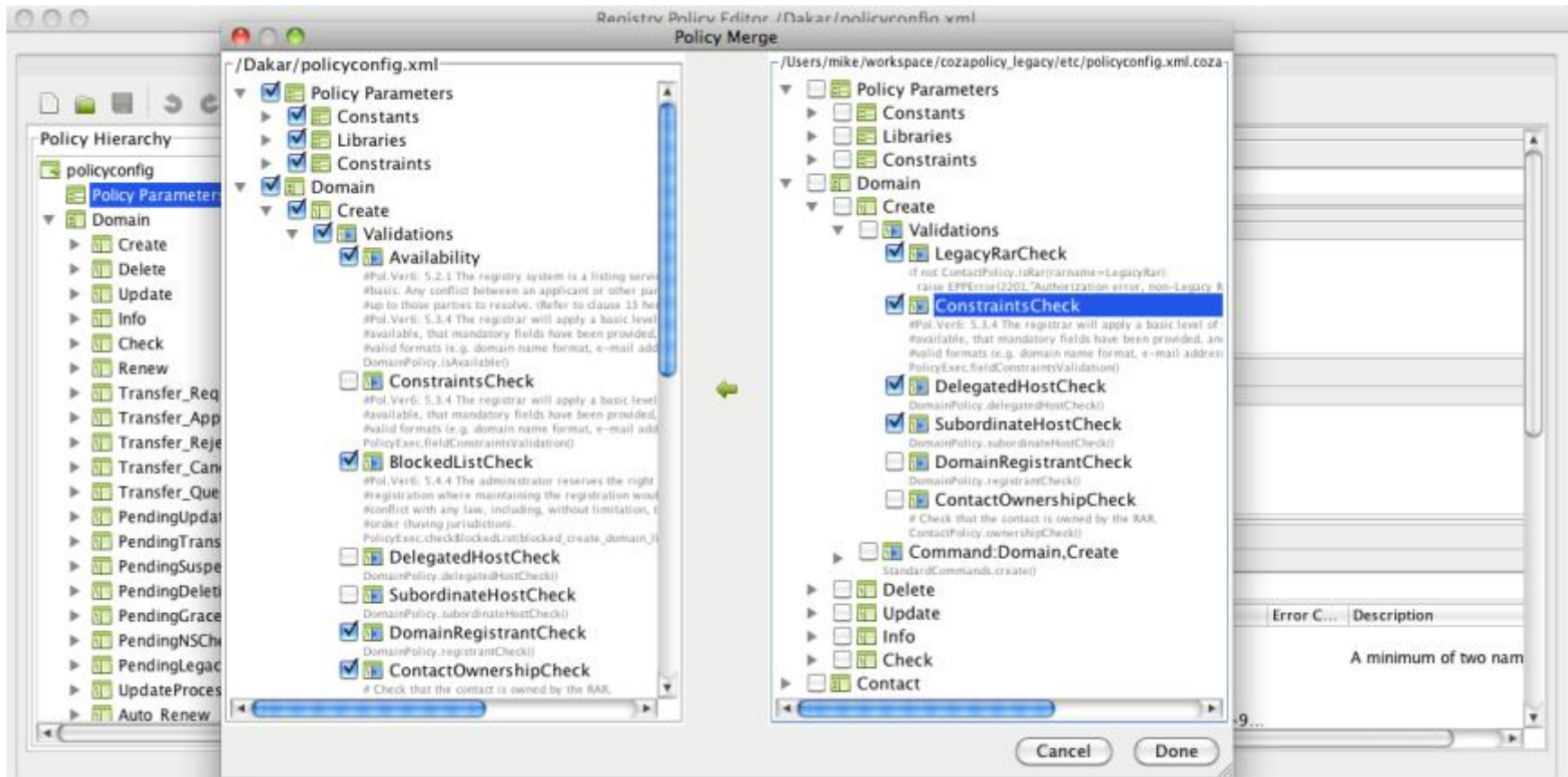


```
<pc:events pc:name="Delete">
  <pc:activities pc:name="ConstraintsCheck">
    <pc:rules>PolicyExec.fieldConstraintsValidation()</pc:rules>
  </pc:activities>
  <pc:activities pc:name="DomainStatusCheck">
    <pc:rules>#Pol.Ver6: 6.2.5 Domain names that are locked cannot be cancelled, suspended or deleted.</pc:rules>
    <pc:rules># Note: We have not branched yet so both apply.</pc:rules>
    <pc:rules>DomainPolicy.statusCheck
      ([&quot;pendingDelete&quot;,&quot;pendingUpdate&quot;,&quot;pendingTransfer&quot;,&quot;serverDeleteProhibited&quot;])</pc:rules>
  </pc:activities>
  <pc:activities pc:name="DependencyCheck">
    <pc:rules>DomainPolicy.dependencyCheck()</pc:rules>
  </pc:activities>
  <pc:activities pc:name="DomainOwnershipCheck">
    <pc:rules>DomainPolicy.ownershipCheck()</pc:rules>
  </pc:activities>
  <pc:activities pc:name="Command:Domain,Delete,DryRun">
    <pc:rules>StandardCommands.delete(dryRun = True)</pc:rules>
  <pc:children pc:name="GracePeriodCheck">
    <pc:branch pc:decision="DomainPolicy.isInGracePeriod(PolicyPeriod)">
      <pc:answerTrue pc:name="InGracePeriod">
        <pc:children pc:name="DomainState:PendingRelease">
          </pc:children>
        </pc:branch>
      </pc:children>
    </pc:children>
  </pc:activities>
</pc:events>
```





# Policy Merge



Registry Policy Editor (Dakar/policyconfig.xml)  
Policy Merge

/Dakar/policyconfig.xml

- Policy Parameters
- Constants
- Libraries
- Constraints
- Domain
  - Create
  - Validations
    - Availability
      - #Pol.Ver0: 5.2.1 The registry system is a listing service. Any conflict between an applicant or other party to those parties to resolve. (Refer to clause 13 here)
      - #Pol.Ver0: 5.3.4 The registrar will apply a basic level of availability, that mandatory fields have been provided, #valid formats (e.g. domain name format, e-mail address)
    - ConstraintsCheck
      - #Pol.Ver0: 5.3.4 The registrar will apply a basic level of availability, that mandatory fields have been provided, #valid formats (e.g. domain name format, e-mail address)
    - BlockedListCheck
      - #Pol.Ver0: 5.4.4 The administrator reserves the right of registration where maintaining the registration would conflict with any law, including, without limitation, the order (having jurisdiction).
      - PolicyExec.checkBlockedList(blocked\_create\_domain)
    - DelegatedHostCheck
      - DomainPolicy.delegatedHostCheck()
    - SubordinateHostCheck
      - DomainPolicy.subordinateHostCheck()
    - DomainRegistrantCheck
      - DomainPolicy.registrantCheck()
    - ContactOwnershipCheck
      - # Check that the contact is owned by the RAR.

/Users/mike/workspace/cozapolicy\_legacy/etc/policyconfig.xml.coza

- Policy Parameters
- Constants
- Libraries
- Constraints
- Domain
  - Create
  - Validations
    - LegacyRarCheck
      - if not ContactPolicy.isRar(rarname=LegacyRar):  
raise EPPError(201, "Authorization error, non-Legacy RAR")
    - ConstraintsCheck
      - #Pol.Ver0: 5.3.4 The registrar will apply a basic level of availability, that mandatory fields have been provided, #valid formats (e.g. domain name format, e-mail address)
      - PolicyExec.fieldConstraintsValidation()
    - DelegatedHostCheck
      - DomainPolicy.delegatedHostCheck()
    - SubordinateHostCheck
      - DomainPolicy.subordinateHostCheck()
    - DomainRegistrantCheck
      - DomainPolicy.registrantCheck()
    - ContactOwnershipCheck
      - # Check that the contact is owned by the RAR.  
ContactPolicy.ownershipCheck()
    - Command.Domain.Create
      - StandardCommands.create()
  - Delete
  - Update
  - Info
  - Check
  - Contact

Error C...	Description
	A minimum of two nam

Cancel Done





# Registry Test Editor



Registry Test Editor /Dakar/domain\_create\_delete.xml

Registry Policy Editor Registry Test Editor

Test Cases

- domain\_create\_delete.xml
  - contactCreate **firststrar**  
Create a basic contact
    - 1000 >> next
    - 2302 >> next
  - domainCreate **firststrar**  
Create a domain
    - 2302 >> domainRenew
    - 1000 >> domainRenew ~ %(period)s
  - domainRenew **firststrar**
    - 1001 >> exit ~ 0
  - deleteDomain01 **firststrar**  
Attempts domain delete with clientDeleteProhi...
    - 1001 >> exit ~ 15

Case

Request Details

None

Poll Request

EPP Command

Details

Template: create\_domain Namespace: domain

Set Fields

XPath	Value
name	%(domain)s
//domain:create/domain:ns/domain:hostAttr[1]/dom...	ns1.rardev.co.za
//domain:create/domain:ns/domain:hostAttr[2]/dom...	ns2.rardev.co.za
//domain:create/domain:authinfo/domain:pw	passwd
//domain:create/domain:registrant	testCont



# Automated Accreditation



## Define Checks

```
{  
  "//contact:check":"//epp:result[@code='1000']",  
  "//contact:create":"//epp:result[@code='1000']",  
  ...  
}
```

## Result

```
registrar - 28/28 passed  
  //contact:check - True  
  //contact:create - True
```

...



# Standards



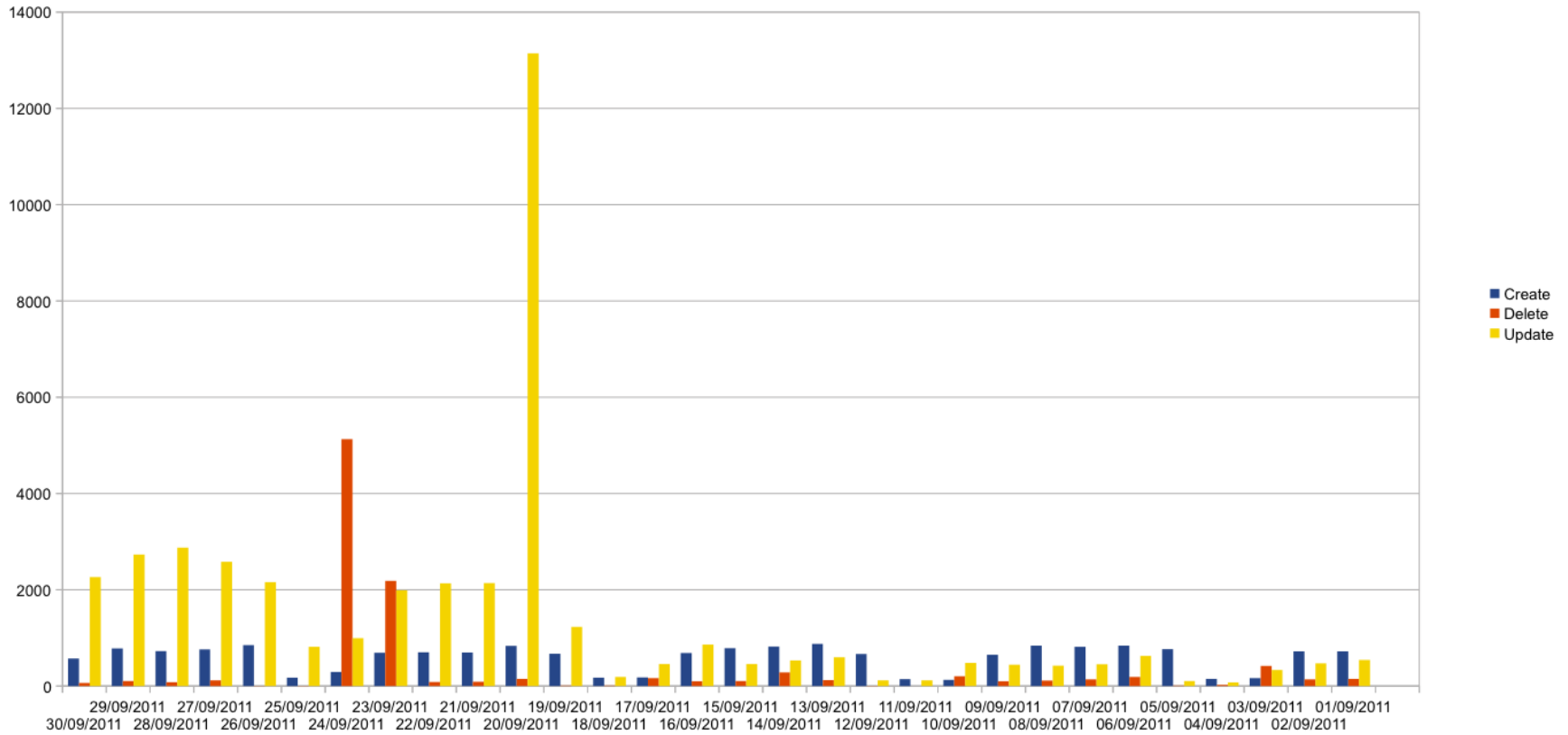
- IETF Std 69 (EPP)
- RFC 5730-4 (EPP, Domain, Contact, Host, TCP/IP)
- RFC 5910 (EPP DNSSEC)
- RFC 3912 (Whois)
- RFC 1035 (DNS Zone File)



# .co.za Statistics



September 2011



# .co.za Statistics



## Average response times for September

<b>Create</b>	<b>70ms</b>
<b>Delete</b>	<b>72ms</b>
<b>Update</b>	<b>51ms</b>



# Q & A



**Presented by**  
**Koffi Fabrice Djossou and David Peall**

**[info@africanregistry.net](mailto:info@africanregistry.net)**

**[www.africanregistry.net](http://www.africanregistry.net)**

