# KC Feedback to Dev Team

## Introduce

* Easy extension of service generation steps. We do have users that want to create their own custom services. Should probably be a simpler extension model than the current. Examples: Persistent systems adding additional operations to data services, imaging workspace that wants to do same. Example: wsenum extension. Example of create extension, etc. John’s recommendation… class on extending Introduce.
* Standards/specs for what Introduce takes as input and what it provides as output. This would help people use the Introduce engine standalone. For the big picture, the Introduce engine could be used as part of other application development frameworks. (Related to above)
* Another example is our Introduce extensions… if the functionality is not present (e.g., no GME), then it should have some decent “failsafe” mode. (TBD)
* Service interfaces. E.g., monitoring interface, metrics interface, etc. (relates to being able to automatically specify monitoring rules and even automatically collect metrics). Related to Introduce, we can “type” collections of operations, etc. Creation of new operations to allow remote collection of metrics. (WSDL limitation). OperationProvider. Define interface and hook it into service.
* Ivy support in Introduce-generated services. Will be solved in 2.0.

## Integration Issues

* Jar issues. We need to be careful about packaging up “big” jars. We commonly run into jar conflict issues when integrating with other enterprise frameworks (J2EE, Web frameworks such as Liferay, etc.). It might be very helpful to distribute a tool to run a classpath check for conflicts. Will this problem be mitigated with new CXF or still to early to tell? (e.g., part of the issue is Globus libraries conflicting with other existing libraries… if we have base networking libs, etc. that are in our own packages, then we won’t conflict so often with other application frameworks).
* Take a careful look at deploying caGrid into existing containers/environments. Should be careful to stand alongside existing apps (e.g., existing applications in JBoss). COPPA use case. Could work with BDA to identify jar conflicts. Maybe do this for common enterprise environments.

## Architectural Issues

* Standard operations for adding data and updating data on data services. Comes from the imaging workspace. Notes: this depends on data model. Need to add an object into a graph makes this difficult from the Grid point of view. Maybe revisit after Identifiers. Could work with imaging workspace. Look at specific ways and see if it can be generalized.
* Standards/specs for other services. E.g., Mayo and incomplete user metadata in LDAP that is used for the authentication service. Should be a very concise (and automated) check for what really are service requirements (clear check on inputs). May also want sequence diagrams of operations to specify common service interactions on the Grid. ECCF and behavioral interoperability. Two separate issues. Other issue is implementation. Question: will the caGrid authentication service work with my LDAP?
* Authentication service removal should be possible. E.g., Emory authentication service going “bad” and causing errors for all users of Dorian. We should anticipate problems between services moving forward.
* Network connectivity. EPRs and Apache URL re-writing and proxies. Message-level encryption instead of point-to-point https.
* At a higher level, re-deployment and migration of a complete Grid (e.g., CVRG use case). This relates to an “offline installation” request. Should be able to re-configure a deployed Grid service even after it’s deployed. Should be able to re-configure an existing container (retrieve settings, modify as needed, and apply). Service migration is smaller and probably the more common use case.
* Grid of Grids and Grid federation models. E.g., Grid directory service… how to do this in Grid of Grids? Monitoring multiple Grids.
* (CVRG) High performance data services and/or move analytical service to the data. Basically, ability to move Grid nodes around for optimal performance. Recommendations on improving model and exposing summary data.
* Integrated authorization provisioning for Grid services. Grid Grouper is the primary example… but services need a standard way to expose capabilities to set authorization on. E.g., even if you have grid grouper, how can you specify who is allows to, e.g., search for Dorian IdP accounts or create stems for a specific stem, etc.

## Technical Issues

* Improved serialization. Need support for abstract types out of the box, improved serialization performance and memory usage (e.g., StAX). E.g., complete customization of binding options (e.g., name of class that maps to QName to avoid problems with Axis defaults)
* Installer issues. Port configuration, maintenance of installer, “browse host credentials on the filesystem”, etc.
* RODS: Maven integration in Introduce. We’ll use Ivy to talk to Maven repositories.
* Formalize link between domain model and semantic metadata: <http://gforge.nci.nih.gov/tracker/?func=detail&atid=174&aid=21704&group_id=25> . Discussed. TBD.
* Offline operation. E.g., compilation looks for schemas online (installer’s container configuration doesn’t work offline). Gather information and file bug. John wants QA test case.
* CDS Delegation. Delegation to service URL instead of service identity (relates to MSI “dynamic delegation” requirements). Not ok. Idea is that delegating a credential is MUCH more sensitive than contacting services. It should be a requirement that the identity is known in advance when delegating a credential. Of course, applications CAN do this if they want to take on the security risks.
* **Identifiers**. CVRG needs a federated view of a subject record. For example, CVRG has both SNP and ECG data for each subject. They would like to present a view of the subject that includes both data sets. We note that these data sets are in separate data services (separate models). They had some questions about how to retrieve these data sets for a subject. The answer is multiple CQL queries to retrieve each data set individually. This could be greatly enhanced with identifiers (identifier that points to complete patient record). I’ll ask CVRG if they want to test identifiers.
* **Data-level authorization**. From CVRG… need comprehensive data-level authorization for all data services. XML data services included. Appropriate to address this in core? TBD.
* **Grid Grouper**. Need to “opt-in” to stems? Photo sharing service use case.

### Service registration

* Goal: identify in advance any registration issues. E.g., firewall configuration is bad. Need to move registration from a service feature to a systems administrator/deployment option.
* Need completely automated service registration and debugging tool. i.e., automated version of the troubleshooting page: <http://cagrid.org/display/knowledgebase/Troubleshoot+Index+Service+Registration>
* Need improved Index service that can detect bad registrations (e.g., registering a service with endpoint “localhost”, non-routable IP, etc.). Immediate rejection of non-routable hostnames in order to display error in container log. Might want to modify existing registration code to make synchronous check first.
* We probably want registration added as a “service module”. What we really need is service registration independent of the service. This module has one configuration file specifying services that it needs to register. We can centrally manage this on a container basis. Take a look at diagnostics service (expanded version of the servlet) that checks registration.
* Really, need remote automated debugging (from the portal) of service registration issues. Self-help portlet that results in “fix X”. Note: need on portal because of DNS issues. (overlaps with metadata updating feature request for portal)

## Usability

* CQL inheritance support (query superclass and retrieve all subclasses)

## Deployment

* IT shops ALWAYS deploy to one system and then move to production after testing. Typically this means moving a VM and re-configuring for production. We need to support easy re-configuration of services similar to what we do with “ant configure”, etc. This does mean easy deployment and configuration of services in various deployment scenarios (port open, apache proxy, etc.). An example is migrating PosSci Grid services when they have completed testing against the Training Grid.
* Cloud model (remote deployment). Nimbus project out of Globus… scripts to contextualize a VM after instantiating a container.
* SyncGTS is messy at this point in time. Most users don’t seem to know what it does or want to install it. We need something that ties into the target Grid and just does the right thing. Maybe every service will sync if needed. Maybe we finally implement remote validation (RSRV) (get rid of syncgts altogether). 2.0.
* Deployment-time checks for service properties, other requirements (e.g., Transfer in container). Admin service check. Transfer extension deployment hook.
* People are very confused about what it means to select a target Grid and how (or if) that affects their existing service. De-coupling registration would fix this problem. Get more specifics.

## Grid Management

GAARDS to evolve into a caGrid Management UI in 2.0. This would encompass:

* Security
* Monitoring (JVM stats from JMX, requests, throughput...)
* Node configuration (Service Metadata editing, target grid change, view container config: port, protocol, hostname, credential location...)
* Registration troubleshooting: automation of registration troubleshooting guide

## Code/Community Projects

* caCORE SDK Type Mapping solution into core? Merge with existing caCORE SDK extension (point at XMI). Could be for 1.4.
* Transfer and soft references bug.
* EPR creation bug… hostname. Is in HEAD/1.4. We’ll do this.
* openMDR into caGrid?

## Community

* Use users mailing list more. Gathering feedback is difficult.