OGSA-WG Interim Meeting #14 — 5 April 2006 — Day 2, morning

Participants

Hiro Kishimoto (Fujitsu)

Andreas Savva (Fujitsu)

Fred Maciel (Hitachi)

Michel Drescher (Fujitsu)

Allen Luniewski (IBM)

Steven Newhouse (OMII)

Duane Merrill (UVa)

Mark Morgan (UVa)

Jay Unger (IBM)

Jem Treadwell (HP)

Marvin Theimer (MS)

Chris Smith (Platform)

Manuel Pereira (IBM)

Ravi Subramaniam (Intel)

Steve McGough (Imperial)

Bridge

Andrew Grimshaw (UVa)

Marty Humphreys (UVa)

Chris Jordan (SDSC)

Frank Siebelist (ANL)

Notes: Jem Treadwell

HPC Use Cases – Marvin Theimer leads

Comments on document:

Steven: Checked on what you *weren't* covering – didn't see any problems, but others might. Marvin: Most comments seemed to be about what isn't covered – trying to cover the simplest – can imagine that they will be covered in extensions. Hiro: In OGSA terminology these would be more solutions for scenarios rather than use cases. Marvin: OK. EMS scenarios are focused on technology behind it; as a first step I think it's important to have the perspective from the back. Hiro: Template provided by Ravi is more for required functionality – e.g. we have requirements in the architecture doc. Marvin: Agree; next step would be to have user requirements. OGSA use cases tend to be very inclusive … they try and cover as much territory as possible, so you get requirements that include everything. Looking for a well-defined method for extension – future versions – OGSA uses cases wanted a broad taxonomy for a horizontal slice, but this is intended to be vertical – what's left out is important, but can be an extension to the base case. Steven: So scenario is a user who has a job described … Marvin: May go to different schedulers … Steven: same virtualized interface – yes… Marvin: My model of states is that there are all kinds of things you might be doing when running – need to define extension to allow a smarter client who recognizes extension to know that a job is doing data staging – need a conservative approach to extensions. Then how do extensions

map to JSDL, BES etc. But extensions are complex, so important to understand the differences. People are clear that too general an extension model is a nightmare – can destroy any chance of it being sensible when composing different sub-states. Steven: complicated state diagram – but implementation doesn't have to do anything as you pass through the diagram. Marvin: Yes. Have to understand how extension works. Andrew: Yes, challenge is to … Marvin: yes, need to acknowledge extensions from the beginning … Andrew: when you have two state models that don't map one-to-one … people will have to recognize that mappings will be lossy transforms… Marvin: people who stay at simple level must get what they expect, so loss can only go one way. Andrew: BES as example, want 3-state model. Given that model, can I take every state in the current BES model and map it to one of those three states. Think the answer is yes, but it is lossy. Marvin: When we look at all the extensions, do we have a practical and reasonable design? Common cases represent things we want to achieve.

Hiro: Prefer base case to common case (parallel/MPI)– Marvin: challenge is that it introduces a variety of things that current schedulers don't support. Parallel jobs are very complex, and putting it in the base case means that every scheduler has to support it. Andrew: Agree with Marvin – doesn't matter at that level of abstraction – just need to take a job restriction, and if it can't do it (e.g. MPI) then it rejects it. May need an attribute on a container that says I can do MPI. Base case should be sequential jobs.

Steven: Confused as to why the ability to support parallel jobs needs to be exposed – not writing a scheduler, defining an interface to access schedulers. Marvin: if I can support parallel jobs then everybody who plays in this space has to support it. Steven: The ability to go from the base case to the above-base case – do you see a change to the interface? Marvin: no, agree it's an attribute of the job. … Steven: Agree – most should be opaque. Marvin: Lot of parties out there who are interested in the degenerate grid case… base case is submit with the additional complexities of going cross-organization, but for someone who's looking at HPC and then grid, 90% are not at the grid case yet – want that community to buy into this with the minimum of extra tax; otherwise developers will fight it. Goal is a seamless design that allows the degenerate case to be handled and carry it transparently to the grid case. Steven: Separation of HPC and grid is good. Marvin: Yes, easier to move someone to version 2 once they've bought into the spec in the first place. Steven: Agree – low-hanging fruit is easier for people to buy into it – this is the right approach!

Hiro: data staging… Steve: don't need much data staging; Mark: UVa does a lot of data staging. Andrew: BES common use case is stage-in/execute/stage-out… JSDL modeled that way… don't think anyone would say copy-in/copy-out is ideal. Steven: In HPC use case I would expect users to have accounts and storage federated in, so all they want to do is put the pieces together that already exist on the resources they want to use. Marvin: Intent to describe cases that users will commonly experience – can give you examples in real life where data doesn't have to move, and there are plenty of cases where it does have to move – significant number of users who don't need to move data. User needs to see a system that doesn't care if no data – staging can be a null stage, or something more decomposed. Getting into mechanism, and the goal is to identify cases we have to deal with – one of them doesn't involve data staging; definitely part of a common case. Hiro: Not binary – what is your criteria? Data staging is common for schedulers – Marvin: common, but not in all of them. Steven: Pre-req work can be transparent to user, and also to service interface – that's the distinguishing thing between HPC and grid cases. [Some discussion missed] Marvin: Most schedulers do support it, but layer on top of core – danger of requiring it is that you exclude anyone who doesn't go that direction. Point of base case is to identify things that are so universal that everyone will buy into it – not all will do data staging.

Andrew: Data staging seems data-independent – typically using gridftp to copy something from somewhere – other parts of JSDL are more specific. Marvin: Need something that GGF & EGA will be able to have a compliance suite for – need to implement – might be able to describe specific case, but once you have to specify a compliance suite then it's a slippery path if things are system-specific. **ACTION ITEM:** Marvin to send e-mail that explains concerns.

Hiro: OK if data staging shouldn't be in base case, but base case/common case criteria is drawn not only from the user's point of view, but from the implementer's. Marvin: Base case is "what is the minimum thing we can come up with", so everyone in the world will have to implement it. Chris: Most Platform customers don't stage with our facilities, so notion of staging is decoupled from the job. We're trying to standardize existing systems, and it's common not to stage data. [Discussion missed] Marvin: In a couple of years probably all would support it, but to require it increases the entry cost; want to avoid that. Want to make it

easy – composition doesn't necessarily cost anything – data staging extension can be implemented by those who need it. Throwing it into the base case costs me something, and the only advantage is that everyone will be able to implement it, but it's not needed.

Hiro: I tend to think from the user's point of view, but agree that we should also consider the developer. Duane: We may benefit by splitting it up. Marvin: No doubt that data staging will be part of what we provide right away. But will still be able to interoperate with more restricted systems. Hiro: I understand, and would like you to document the criteria.

Marvin: Do these scenarios cover the interesting cases? Notification vs. polling – one common case is to submit an RPC and the job is done when the RPC returns. Should this be a common use case? Any others? Steven: Guidance should be to limit.

Steven: What's the next stage? Marvin: Once we agree convergence I'll ask in the mailing list about architectural implications of taking scenarios and mapping them to design – need strawman proposals. One path is to take things that exist – e.g. JSDL & BES – see how they fit. And/or work out how extension will work. Also CDDLM, RSS. By GGF17, would like some serious strawmen out there, to be picked apart. Prefer to do it mostly by e-mail, rather than telecon. Hiro: Would like to do one telecon on this. Steven: Would like to look at deltas needed to BES and JSDL rahter than start a new interface definition activity.

Next document: Intermediate document: use case with requirements description, then strawman design (high-level design, no XML), hopefully make progress before GGF17. Will produce as GGF-format documents (but there's no restriction on the format for the body).

Marvin has ideas for whom to talk to for each type of scheduler – interested in more contacts.

Chris: Parallel jobs are not base case – is that true for HPC? Marvin: I'd argue that it's not base, but common. Chris: Would like to distinguish between multi-CPU jobs and parallel jobs. e.g. 16 CPUs is still a simple case. Failure more would be job failure, not node failure. Multiple CPUs on an SMP, and multiple CPUs on a cluster. SMP is very common. Might be an extension that's defined right away. Marvin: would prefer to keep it out of the base case. Chris: that's fine, but would push to do the extension right away. All iterative; will evolve!

Hiro: WS-Man-based OGSA basic profile previously tabled – what is the situation now that the Roadmap is out? Marvin: Reconciliation implies that we don't need the second stack, so that whole effort could be put on hold or disbanded because there will be a new system management design. WS-Man & WSDM look like they'll converge – broader community will harmonize based on good aspects of each. We need to participate in that rather than go against it. Hiro: But it may take two years to do it, and your product will be shipped in that time. Steven: What happens to WSRF profiles? Valid now, but will be redundant in the future as we move to the converged world. Do we need an OGSA WS-Transfer Basic Profile? Marvin: Not a whole lot of system management needed at job scheduling level … Jay/Hiro disagree – need to handle state in a standard way. Steven: How can there be a standard way of handling state in job submission… standard mechanisms that can be used, but no standard way to do it. Hiro: Using standard mechanism is best way to guarantee interoperability. Jay: IBM pushing to use standards for marshalling… can't legislate that a job will have a same state map as a disk drive, but tooling can be built around notion that there are ways to represent them. Where does system management begin/end? Marvin: No problem having a system management interface as well as a scheduling interface… can argue about how much to standardize, but at the scheduling level I can wait for harmonization or I can expose an interface that allows a client to submit a job and get its state – clients don't care about system management. If there was a standard pattern I would say use it, since it's there, but absent that I would add the functionality, and move away from it once we have standards. Jay: Disagree: WSRF is a standard now, and should continue to be used. [Discussion] Roadmap says key specs will be published quickly, so base standard is not expected to take long to emerge (2-3 months?), though it may take a while to go through the process. This is a case to do the parallel profile work. Don't want to stop working on standardized stateful operation. If I want to re-use code between applications, I do care about standardizing state operations. Web services is not just another RPC invocation. Chris: People are doing well now – not using standard state representations, but doing ok based on WSDL. Ravi: Semantic level and how do we set it up; many ways to represent and manipulate, but semantically they're equivalent. Just proposing the semantics? Marvin: I don't think we're 2-3 months away from understanding it all – lots of open questions. Lots of things in GGF that haven't been

harmonized with DMTF. Don't think we'll have harmonization on details this year. It will be a few more years before you can just plug something into the system – would rather decouple myself for now; agree with Jay, but skeptical about when it's going to be done. Jay: Difference between complying with WSRF and defining your own WSDL? Either will be superseded, so you'll have to change anyway. Promise of functional equivalence between final standard and what's available today, so easier to fall back on what exists.

Steven: Should schedule a session at the f2f following publication to determine if we need to do a new OGSA profile. Jay: Given that, why choose a third representation? Marvin: Today there's no lifetime in WS-Man; not going to use WSRF.

Discussion end (out of time).

WS-Naming

Objective: Close out trackers where possible, so we can take the document to public comment.

Tracker 1738: Andreas: Are you going to publish the use case (background) document as a GGF document? May be in the GGF Roadmap – Mark/Marvin will find out Andrew's intentions. May need to be updated to cover these use cases. Can't close this tracker; change state to Pending.

Tracker 1739: Concerns about uniqueness of generated abstract names. Mark: Want to recommend, not specify. Jay: Tom's concern is that it needs to be specified. Marvin: Andrew's comments apply if I don't care about auditable uniqueness. If I need to audit it, it's a much stronger use case, and the requirement to specify applies. Not sure if we need to address that. Jay: Need to be careful about the scope of uniqueness within a service, because you don't know how services will be composed. [Discussion re auditing] Mark: Purpose is not for cryptographic integrity. Hiro: Happy with Andrew's response re uniqueness, but doesn't respond to second part. Mark: Tannenbaum's book says you cannot use an address for an identity, because it may change. Hiro: Need to add to the tracker.

Naming document has been updated, but no notification sent to the mailing list,. No discussion possible until people have had a chance to review the trackers and current document.

WS-Directory

Mark presents – see slides.

This is a map of human-readable name to EPRs – not a resolver.

Manuel gives some background and status of RNS. Submitted to GFSG, but been pushed back. Consider refactoring RNS and submitting in filesystem space, but concerned about being usurped. Mark: Proposed WS-Directory to solve an immediate need at UVa, but if a refactored RNS solves the problem then we don't need it. Manuel: RNS has been distilled further & further – it's essentially just a registry, and can be aggregated – if it works in an O/S environment then it can be extended to a grid. RNS has had a lot of work, and we'd like to see participation to make it faster, so we can all benefit. If it's too bulky… Comment so n WS-Directory say that it's missing things; being built on. Steven: Could there be a WS-Directory and an RNS spec that builds on that? Manuel: Yes, I think that's reasonable. Chris Jordan: Can do that and preserve the RNS work; lots of valuable concepts, people need to be able to implement just what they need. Manuel: Is there a general agreement that we should work together to refactor this to address the issues? Don't know the specifics – what should be the next step? Mark: From UVa GBG group, not painful for us to go forward with WS-Directory, but don't want to invalidate the good work that's already been done. Manuel explains RNS facilities – notes extensibility. Mark: Can't tell you which way to go; we're happy to make this work as well as we can, but you need a specific list of requirements. I'm happy to provide that list. Chris Jordan: Happy to help. Manuel: Would be happy for WS-Naming to take ownership. Ravi: I would advocate that RNS be adopted; has a richness that's relevant; effort to be spend on refactoring to simplify without diluting it is an activity that we should sponsor. Mark: Flexibility: built-in or not limiting what can be added. Ravi: Also possibility of adding a profile concept. Mark: that even plays into WS-Directory. Manuel: I think composability when I hear refactoring. Hoping we have a vision that an aggregate set of

services would accomplish what we have in the spec. RNS should be factored out and distilled. Mark: Propose that I generate a list of requirements; Manuel to generate a list of key points you want to keep. Chris J: Agree. Hiro: Clarify steps… Mark: I would continue to develop WS-Directory as an illustrative example, and either adopt it or drop it if RNS meets requirements. Use it as a contingency, rather than a competing spec. Manuel: Never seen a list of issues with RNS; I want to invite anyone interested to work with us, rather than competing with us.

Hiro: Notes Greg Newby's recommendation – rename or refactor, and adjust functionality (simplify). Manuel: Yes, considering taking out functionality. Mark: Our hope is that we can throw out WS-Directory, but we don't feel that we can do that now. But it's not a specification; don't want to compete.

Chris Jordan: Would like to see these discussions continued on a regular basis. Will volunteer to coordinate conference calls and track suggestions and status. Manuel: Needs higher visibility, because this is not related to GFS, but to WS-Naming, and I'm not sure that it needs to continue in GFS, but in OGSA. Chris: Agree, but someone needs to volunteer to move it forward. Discussion about where to host discussion. **ACTION ITEM:** Manuel & Andrew to determine where/when to hold discussions.