A HYBRID ARCHITECTURAL STYLE FOR COMPLEX HEALTHCARE SCENARIOS

Leigh Griffin[1], Christopher Foley.[1], and Eamonn de Leastar[1]

[1]Waterford Institute of Technology, Cork Road, Waterford, Ireland.
{lgriffin, ccfoley, edeleastar} @ tssg.org

In classic software engineering, a successful software architecture arises from functional and non-functional requirements analysis, modeling, design elaboration and implementation phases, incorporating key trade-offs and constraints. This paper proposes an alternative approach, informed by deep insights gained from understanding successfully deployed architectural styles in two key domains: highly scalable, resilient web applications; and robust presence and messaging systems. We propose that the challenges and complexities within the healthcare domain can be successfully addressed with this approach. Specifically, the REST architectural style with its focus on resource oriented architecture, and the Jabber protocol set and its associated messaging and presence infrastructure. These two approaches have been successfully implemented on a global scale, have been bound to legacy informations systems, and have demonstrated an ability to evolve to match the most complex organizations. The approaches are complimentary, but not without contradictions. This paper discusses these contradictions and lays out a set of challenges that, if successfully addressed, can yield a flexible, powerful and resilient architecture within a highly challenging domain.

***Index Terms*—REST, healthcare, communications software, distributed systems.**

1 Introduction

Modern hospitals within Ireland face daily problems when it comes to inter-department communications. A simple request for information between departments can suffer considerable latency and be highly variable in terms of quality and delivery. This paper proposes an architecture which could decrease the turnaround in information exchange between departments, offer controlled data sharing within a hospital environment and ultimately improve the quality of care for the patient.

This paper is broken into 7 sections. Section 1 is this introduction. Section 2 describes a typical scenario within an Irish hospital setting. Section 3 "takes a systems of systems" view on the hospital environment from a REST perspective. Section 4 examines a means for facilitating communications through XMPP. Section 5 looks at the application of REST and XMPP principles to the scenario. Section 6 looks at the challenges that such a hybrid approach raises. Section 7 is the conclusion.

2.1 Scenario

A patient arrives at a hospital A&E. The receptionist looks up the patients A number (everyone who has ever been in this particular hospital has this number) and creates an admission document, notifying the triage nurse within the A&E that another patient has arrived. A request is sent to the archives department to retrieve the paper version of the patients known medical history chart.

When it is the patients turn, the triage nurse assesses the patient and prioritises the patients care. A temporary chart is used (while the full medical history chart is requested) and initial vitals and visual assessment notes are added. The triage nurse makes the report and depending on the priority assigned the patient will be seen immediately or in turn. The report is left in the on duty doctors drop box.

The doctor, when available, takes the patients report and sees the patient. This report will form the basis of the doctors observations and if the doctor deems it necessary the patient is admitted to the hospital. A ward most appropriate to their current condition is selected and the patient assigned to it. The doctor makes a phone call to ensure this ward has the capacity for another patient, and if it does, a porter is called to transfer the patient. The current matron in charge of the ward at the present shift time is charged with admitting the patient upon receipt from the porters. Finally the doctor adds to the triage nurses report requesting tests ranging from Blood Pressure, to Urine analysis to toxology reports to be completed upon admission to the ward. This chart along with the patient are transferred by the porters to the ward.

On admittance to the ward, the matron is responsible for allocating a bed to the patient depending on factors such as medical insurance and need for privacy or isolation. A doctor is assigned, by the matron, to the patient. The doctor is notified immediately either through a desk phone, a beeper or a mobile phone depending on factors such as the urgency of the case and his location within the building. The matron at this point in time has received the full medical history from the archives department and the matron combines the temporary A&E charts with the full medical history. A ward nurse is assigned the tests requested by the admitting doctor and the matron creates further tests that are deemed relevant based on the full medical history that is now made available.

The ward nurse reads the A&E chart and knows the tests that need to be performed. All test data is recorded on the patients chart and the samples taken are labeled and where appropriate tests are completed on the spot and the nurse records this data. The nurse contacts the matron and the doctor immediately if

any abnormal results appeared in the initial testing. The rest of the samples are taken to the lab for analysis by the nurse. The technician responsible for these results is notified of the urgency of the case and when the results are available, they are faxed or rang through to the nurse who requested the testing.
The doctor, upon arrival can see the reports and patients history chart which together gives a detailed account of the patients current state and past history respectively. After a first hand investigation of the patient, a prognosis is made and a care plan assembled. The doctor communicates this care plan to the matron nurse who is responsible for assigning each shift the tasks outlined in the care plan. At the end of every shift a handover occurs whereby a group meeting between the ending shift and the next starting shift happens. Doctors and nurses on the ward are required to attend as the transition is important for the well being of the patient and the execution of the care plan. This handover occurs at the end of every 12 hour shift.

When the patient is discharged from the ward, a final report is created and added to the patients medical history and the archives department notified that the updated chart is ready to be collected.

2.2 Use Case Model

In Figure 1 we identify the actors involved, and the interactions that occur within use cases identified in this scenario. In Section 2.3, we take two interacting use cases, *Preliminary Investigation* and the *Test Request,* expanding on them with a technological perspective. These use cases are described in an idealised environment equipped with a suitable software architecture. This architecture is elaborated in the remainder of the paper.

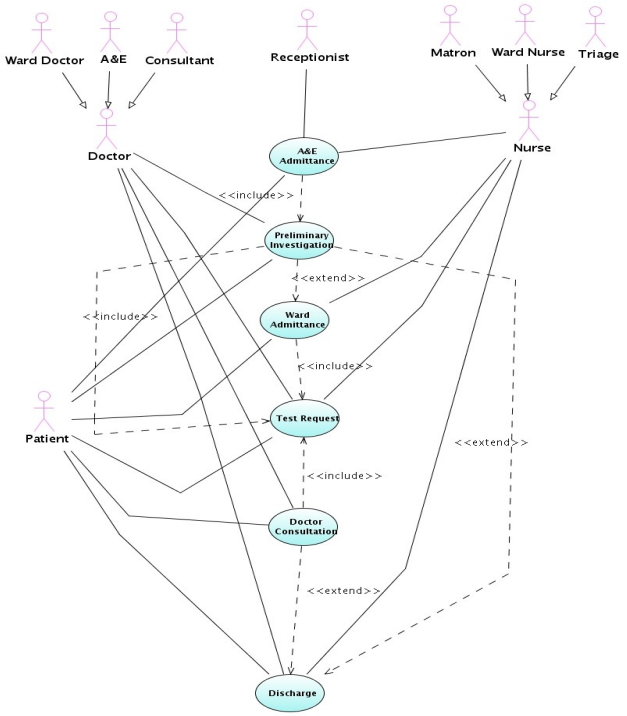

Figure 1 Use Case Model

2. Expanded Use Case

The doctor receives a communication on his PDA informing him that a patient has been allocated to his queue and that he has been added to a new *care group* for this patient – an active list of the healthcare professionals dealing with the patient. A flag appears next to the group showing how urgent the case is. Depending on the seriousness the doctor will see the patient immediately or when the patients turn in the queue arrives.

When ready, the doctor takes out his PDA and checks the contents of the care group. He notes two files, the patients full medical history, and his temporary chart containing a short detailed overview from the triage nurse that attended the patient already. He opts to read the short overview as the detailed medical history would be overwhelming at this point.

The patient is seen and assessed. The doctor requests that a nurse come along and take the patients blood pressure and glucose level. The doctor in the meantime goes to treat an urgent case. Upon return the doctor finds in the care group, test results that the nurse has uploaded. The doctor makes some notes on his laptop and an initial prognosis and assumption as to the cause of the patients illness is delivered. A decision is taken to keep the patient in for observation and a consultation with a specialist.

The doctor accesses a service showing him the available beds in the ward that he feels matches the patients condition the best. A service request is sent to the matron of the ward requesting the bed and this resource is removed from the available bed's service in case a double booking is made. The matron, who's ward the patient will be placed in, is added to the care group which from now on in will consist of the care team responsible for issuing and following a care plan for the patient. The matron is given the management rights to this group as the principle care administrator, as it is now the matrons role to allocate ward nurses and appropriate consultants to the care team.
The doctor uploads his report and creates several testing services that he feels should be performed upon admission to the ward. These services in turn will be handled and allocated to nursing staff by the matron. Finally the doctor accesses the porters service and sends a request for the patient to be moved from A&E to the ward. The doctor proceeds to the care of the next patient on his list.

3.1 System of Systems View

At one level this scenario seems amenable to a conventional analysis, modelling and implementation based on classic enterprise oriented tools and techniques. For instance, the Rational Unified Process [1], with its six disciplines (Business Modelling, Requirements, Analysis & Design, Implementation, Test and Deployment) could yield a workable model expressed in a variety of UML artefacts, accompanying data dictionary and procedural documentation. Indeed there would probably be a reasonable chance that such an implementation could succeed in a limited context. Provided the scenario as documented is accurate and the environment is highly regulated and controlled, then it may even prove moderately long lived. However, these are very significant assumptions, unlikely to hold firm within a healthcare context.

Re-examining the scenario, on a deeper analysis it could be argued that it is in fact a description of multiple information systems, intersecting often in an ad-hoc and unpredictable fashion. Each system is evolving independently of the other, undergoing differing constraints and limitations, and most likely underwritten by substantial legacy application infrastructure.

This is in fact a "system of systems", each of which encompasses its own concerns, socio-technical practices and independent development cycle. Such systems are among the most challenging to analyse, model and implement. Indeed, if one bears in mind the growing importance of pervasive sources of information (sensors and other embedded sources not explored in the scenario) then the problem becomes considerably more complex. The US DoD sponsored Ultra Large Scale Systems (ULS) [2] report charts this territory, outlining a research agenda stretching well into this century. In the ULS analysis, a recurring metaphor is the comparison between a building and a city. Whereas a building is "engineered", amenable to consistent and repeatable best practice in design and construction, a city is considerably more complex. Cities are "not simply bigger systems: they will be interdependent webs of software intensive systems, people, policies, cultures, and economics". These "systems of systems" are radically different from conventional software systems: they comprise "a dynamic community of interdependent and competing organisms (in this case, people, computing devices, and organisations) in a complex and changing environment". Taking a typical hospital and applying this analogy, the hospital can be identified as a system of systems. A hospital is made up of "systems" such as the A&E department, X-Ray department, Surgical wards, Medical wards, theatre, finance department, record department and many more.

Such systems pose serious challenges for traditional enterprise and model-driven approaches - hence the ULS agenda to investigate computation emergence, bio-inspired models, policy driven development and other more exotic disciplines. Whilst many of these disciplines are likely to achieve major breakthroughs (perhaps in a 10-20 year time-scale, the agenda for ULS research), this does not necessarily mean that highly complex systems are beyond the reach of current technology. In fact, there is one highly complex "systems of systems", complete with independently evolving constituent elements, in some instances encapsulating diverse legacy information systems and distributed across multiple organisational, legal and technical boundaries. This is of course the Internet, the most successful networked technology platform yet devised.

While the full stack of internet protocols is engineered to the highest standard, and continuing to evolve to meet new challenges [3], it could be argued that the network has evolved above all into a services platform, with the World Wide Web set of protocols [4] as a foundational specification for a truly global platform. These protocols [5],[6],[7] have proven themselves as resilient, scalable, robust and secure. Critically, they are also comprehensible, easily understood, and embody a set of principles that, if followed, enable a services to be constructed that are in tune with the scalable, independently evolvable and robust nature of the network. It is possible to construct an enterprise application across the network without adhering to its principles [8][9], effectively building an overlay network, re-purposing it with a new architecture. These attempts have not been noticeably successful. However, if the network is used as its designers intended, then it is possible to construct outstandingly successful, integrated yet modular and resilient services. These principles are known as REST - Representational State Transfer [13] - and they encapsulate the current best practice for building complex "systems of systems" .

3.2 REST

Representational State Transfer or REST [13] is a set of design criteria for distributed systems that stress component interaction and scalability. It is not a specific architecture but more a set of principles which one should adhere to make their architecture *RESTful*. It is not dependant upon any particular protocol and REST is not a standard per se. however it does prescribe the use of standards, e.g.: HTTP, URL, XML, HTML, + the full constellation of media formats and standards.

The key principle of REST is the role played by *resources*. A resource is anything that's important enough to be referenced as a thing in itself. REST principles can be summarised as:

- *Uniquely Identified Resources*; The URI is the name and address of a resource. A resource must be addressable.
- *Uniform Interface*; resources should only be accessible via a constrained set of operations and represented with a constraint set of content types
- *Communicate Statelessly*; this means that each request must happen in complete isolation. The request must contain all the necessary information that the receiver needs to process it.
- *Layered System*; resource representations are interconnected using URLs enabling a client to progress through states.
- *Cache*; cached responses are proposed to improve network efficiency
- *Client-Server Pull*; clients pull resource representations from servers.

REST in itself is a high-level style that could be implemented using many different technologies. HTTP is the predominant instantiation of the REST uniform interface. As with all architectural styles, there are advantages and disadvantages to building RESTful architectures. Some advantages

- General uniform interfaces
- Scalable component interactions
- Reduced need for resource discovery as resources are inter-linked
- Not state dependent

Disadvantages

- Multiple client-server requests may impact network performance
- If instantiating a REST architecture using HTTP as a uniform interface raises problems around asynchronous events being transferred to a client

4 Communications View

Reviewing the healthcare scenario again, an architecture to realise key functions can clearly benefit from a RESTful approach. However, there are also features that go beyond the stateless, resource oriented nature of REST, requiring additional core capabilities. These are most apparent in the near real time communications required by the scenario. In the communications, six key elements are important:

- A group (referred to as a care group) is assembled around a specific task
- The current status/location, each member of this group is visible to all of the group members.
- Depending on this status, messages can be exchanged in near real time between group members
- Supplementing messages, files (images in particular) can also be exchanged.
- Such conversations can be carried out on mobile devices
- A log of all message and data traffic is readily available

Historically, this would have been the realm of groupware applications, which have evolved such features over successive generations within the corporate realm [14]. With the growth of the Internet and the strong reliance on open protocols, these capabilities have been incorporated into highly scalable Instant Messaging systems. The most prominent and successful of these is the XMPP set of standards[11], also known as Jabber. Jabber is built on five key principles:

- Network—All Jabber domains that exchange messages. A network must contain at least one domain.
- Domain—A subset of the network containing all entities that handle or belong to a domain. Provide local control over parts of the Jabber network while still communicating with users outside of the Jabber domain.
- Server—A logical entity that manages a Jabber domain.
- User—An entity representing a logical message delivery endpoint. Users are managed on the server with user accounts.
- Resource—An entity representing a particular message delivery endpoint for a user. Jabber IM clients play the role of Jabber resources.

Additionally, the protocol itself is based on fully open XML Schema, with specific extension points clearly denominated. These enable additional schema to be defined and, with appropriate extensions to interested clients, the standardised XMPP servers can seamlessly handle traffic with these extension payloads.

Messaging systems constructed using the XMPP protocols have some interesting similarities with the RESTFul architectures discussed earlier. Both can be viewed as replacements for earlier more heavyweight approaches, both rely on the inherent properties of the underlying network (DNS, TCP/IP protocols, URIs), and both embody relatively simple principles that can be adapted and realised into highly innovative services and applications that can scale dramatically. However, they both also differ substantially in other areas: although they both use the concept of a resource, the semantics of a resource is quite different for each. Whereas REST relies on stateless communication, XMPP maintains a single session to the communications server through which all traffic passes. Finally, and perhaps most interestingly, REST is an architectural style that can be realised using the HTTP set of protocols. XMPP however, can not be regarded as an architectural style as such, but is rather a highly successful protocol set within the near real time communications domain. Part of the challenge of this work is to "reverse engineer" a set of principles from XMPP, and integrate them with the already well formulated REST principles.

4.3 Information Landscape

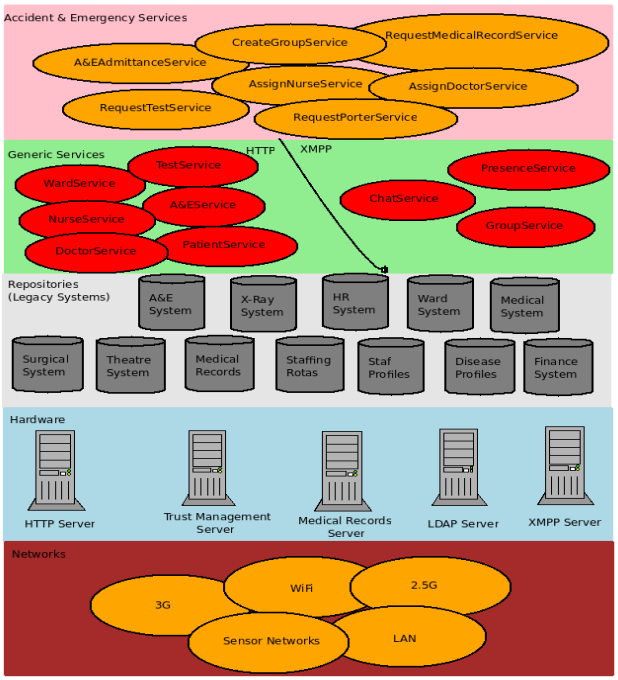

Figure 2 Architectural Layers

This information landscape outlined in the healthcare scenario is challenging and complex. A complete picture is beyond the scope of this work. Figure 2 identifies the broad layers that are

part of the information system covered by the scenario. A key objective is to introduce efficient, secure, non-invasive and scalable overlay on top of legacy information system, that embodies the principles discussed. This overlay should enable new services, new interaction patterns and ordered evolution of existing systems.

5.1 Application of RESTful Principles

This section looks to apply the REST principles described in Section 3 to the expanded scenario specified earlier in Section 2. It also focuses on an XMPP implementation of the REST principles and identifies the added value that this technology brings when coupled with a RESTful approach.

The first task is to identify the main resources within the system which would allow us hold, transfer and distribute key data between the key actors. Figure 3 identifies a sample of the resources which have been identified. A resource is shown to have attributes. The value assigned to a resource attribute may be a URI of another resource thus applying the 'layered' principle of REST, e.g. the attribute temporaryChart in the patient resource has the value of the temporaryChart resource.

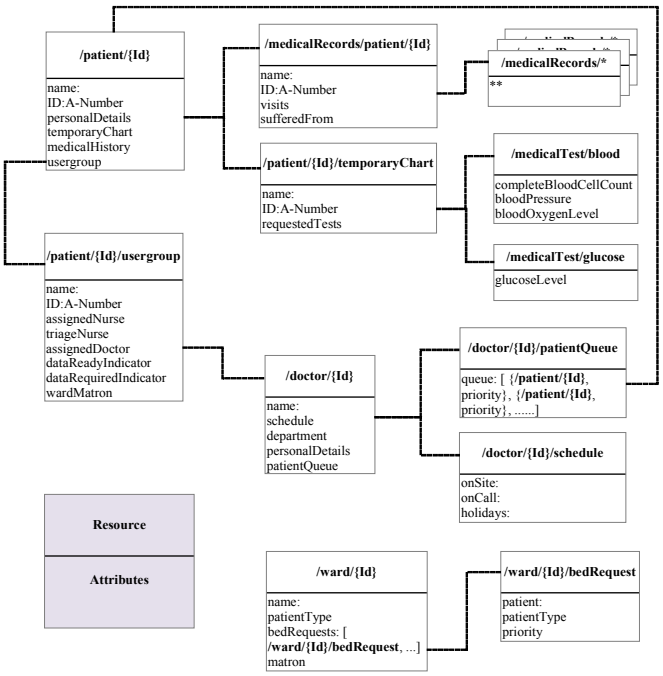

Figure 3 Layered Resources

Figure 4 shows the sequence of events which may occur when the scenario is realised. The TriageNurseService (not visible in Figure 3 for readability reasons) does a PUT(i.e. modify) on the doctor/patientQueue resource which assigns the patient to a doctor with a priority setting. As part of this process a care group is set up for that patient including the patient assigned carers (e.g. nurse and doctor). Once this is set up all members are subscribed to events which occur within this group. This is built in real time functionality which XMPP provides.

The doctor gets the temporaryChart and does the first analysis, which results in ordering of further tests. This ordering is done by performing a PUT on the nurse/patientQueue resource and also performing a PUT on the requestedTests attribute of the patient/{Id}/temporaryChart resource. The Nurse can perform the tests and do a POST (i.e. create new resources) for both medicalTest/blood and medicalTest/glucose. From this point the doctor needs to be notified again (see the Asynch Notification in Fig 4) that the tests are completed and the patient can be reassessed. The care group is updated indicating that data is ready. This will in turn flag to the subscribed users (i.e. doctor and nurse) that the results are ready.

The doctor can then reassess the patient based on the test results. Based on the doctors prognosis, he can create a /ward/{Id}/bedRequest resource for the ward which is appropriate. This request can be handled by matron or ward responsible. The matron is then added to the patient care group, as one of the carers. The care group concept changes personnel (cares) as the patient resource moves throughout the hospital departments (systems of systems).

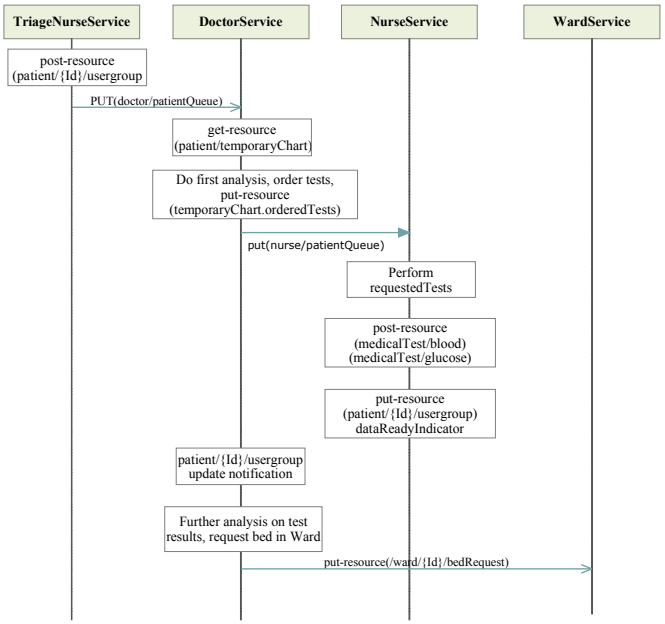

Figure 4 Sequence Chart

Once the patient actually moves from A&E to the ward, then the matron becomes the key driver of the patient's care group. The care group will be modified to exchange the A&E nurse and doctor to ward nurse and doctor as careers for the patient.

5.2 XMPP view

This section looks to apply the key principles of XMPP introduced in Section 4. Taking the systems of systems approach, each system or department within the hospital would be represented as a Jabber domain. A hospital network would be created governing all of these domains allowing inter department communications. Each domain would retain its own autonomy at all times. A Jabber based server would be installed, integrating seamlessly into any existing LDAP based directory for user authentication. Each staff member within the hospital would be registered with the Jabber server and be represented as a User. A desktop computer, a laptop, a PDA or

a mobile phone would be used by a User as a Resource to gain access to the Jabber network.

Taking the above scenario, the doctor receives an instant message on his PDA letting him know that a patient has been allocated to him and directs him to the web services responsible for handling the patient and the data. On the doctors contact list a new care group has been created with the patients name as the group title. After assessing the patient, the doctor wishes for some tests to be performed. On his roster list he checks to see what nurses are listed as available. Those down as busy are already occupied with a patient. When a nurse is located the doctor opens a chat session with her and outlines what he wants done. The nurse is sent a group invitation and on her roster list a new group containing the doctor has appeared. Tasked with these tests, one of which involves an interaction with the lab, the nurse goes and carries out her work. Bringing up the Lab Departments roster she contacts an available technician and outlines what specific tests need to be carried out. This technician is sent a group request and joins the existing members in the roster. When the test results are available, the technician sends a message to the nurse informing her that the results are now available. Upon delivery of this message, the technician has no further need to be in the group and leaves on his own accord.

When the doctor receives the test results and decides that the patient needs to be kept in for observation, he opens a chat with the matron of the destination ward. The matron is added to the patient group and given full admin rights. The doctor finally contacts the porter requesting that the patient be moved. Having now completed all requested actions in relation to the patient, the doctor leaves the patient group and moves on to the next task at hand.

6. Challenges of a hybrid approach

Successfully merging two architectural styles poses some significant challenges from a development perspective. Four challenges for developing this hybrid style have been identified:

- Incorporate RESTful approaches into XMPP. The styles and principles laid down by REST would be brought into XMPP by it's extensible nature.
- URL endpoints chain into XMPP. Looking at the opposite direction, have an implementation of REST used to stimulate XMPP. Services directing to URL endpoints would be used as the trigger for XMPP messages.
- Resources. Both styles pose two different concepts of a resource which were shown in Sections 3 and 4. An XMPP resource would need to be treated as a REST resource or vice versa. A common resource definition and usage is vital for a hybrid model to succeed
- “Stateless-Session” Hybrid. The Stateless world of HTTP and the Session based world of XMPP could be combined to form a hybrid “stateless-session” of sorts

7. Conclusion

Engineering complex systems poses considerable risks. The industry is littered with large scale failures, particularly in publicly funded information systems. The REST architectural style, coupled with an efficient messaging and presence system, has the potential to deliver a robust and resilient foundation for complex information systems. These two approaches are particularly relevant where there is a high priority on group collaboration, significant legacy information systems and a critical, near real time, communications requirement. Both styles are complimentary, but not without contradictions and tensions. This paper proposes both as providing a solid starting point for such a system, a baseline infrastructure, a set of assumptions and a technical vocabulary already imbued with appropriate architectural semantics. The contradictions are also laid out, as a series of challenges, which have the potential, if addressed and resolved, to yield a powerful and resilient hybrid architectural style.

8. References

[1] Mancin, Enrico et al., 2007. The IBM Rational Unified Process for System z. IBM Redbooks.

[2] Ultra Large Scale Systems [online]. Available at http://www.sei.cmu.edu/uls [Accessed on 25-OCT-2008]

[3] Ipv6 [online]. Available at http://www.ipv6.org/ [Accessed on 29-OCT-2008]

[4] IETF & W3C [online]. Available at http://www.w3.org/Signature/ [Accessed on 29-OCT-2008]

[5] Hyper Text Transfer Protocol – HTTP/1.1, [online] Available at http://www.ietf.org/rfc/rfc2068.txt [Accessed on 29-OCT-2008]

[6] Domain Name System Structure and Delegation [online] Available at http://www.isi.edu/in-notes/rfc1591.txt [Accessed on 31-OCT-2008]

[7] XHTML 1.0 The Extensible Hypertext Markup Language (Second Edition) [online] Available at http://www.w3.org/TR/xhtml1/ [Accessed on 19-OCT-2008]

[8] The Corbra programming language. [online] Available at http://cobra-language.com/ [Accessed on 3-NOV-2008]

[9] Web Services at WC3. [online] Available at http://www.w3.org/2002/ws/ [Accessed on 14-NOV-2008]

[11] XMPP [online]. Available at: http://xmpp.org [Accessed on 29-OCT-2008]

[12] XMPP-Extensible Messaging and Presence Protocol: SOAP and REST get closer company [online]. Available at:http://searchsoa.techtarget.com/tip/0,289483,sid26_gci1332820,00.html [Accessed on 15-OCT-2008]

[13] Representational State Transfer (REST). [online] Available at: http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm [Accessed on 1 OCT-2008]

[14] Groove Networks [online] Available at http://www.groove.net/ [Accessed on 14-NOV-2008]